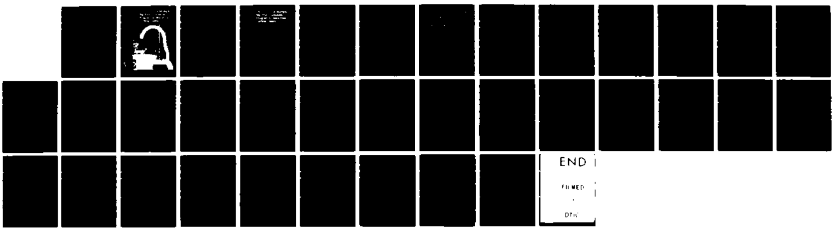
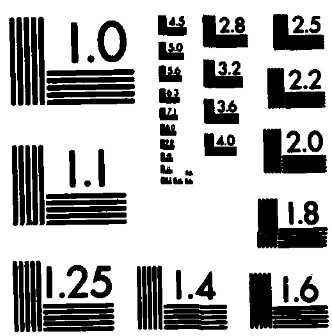
AD-A122 125 OPTYLD - A MULTIPLE RIP-FIRST COMPUTER PROGRAM TO MAXIMIZE CUTTING YIELDS(U) FOREST PRODUCTS LAB MADISON WI P J GIESE ET AL. SEP 82 FSRP-FPL-412 1/1

UNCLASSIFIED F/G 9/2 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

United States
Department of
Agriculture

Forest Service

Forest
Products
Laboratory

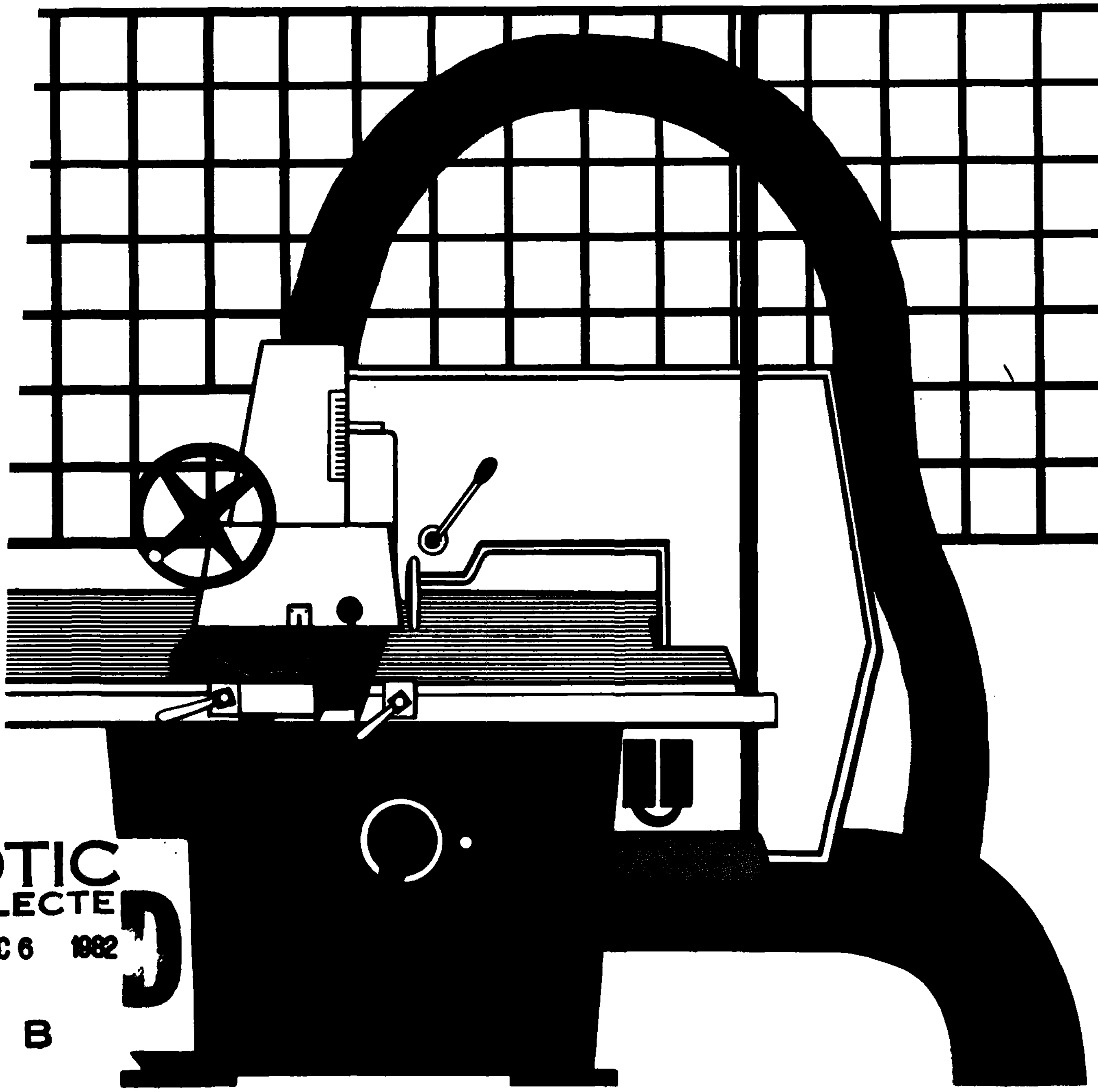
Research
Paper
FPL 412

(12)

OPTYLD — A Multiple Rip-First Computer Program to Maximize Cutting Yields

AD A122125

DTIC FILE COPY



DTIC
ELECTE
DEC 6 1982
S D
B

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

82 12

Abstract

This paper describes the computer program OPTYLD that is designed to simulate multiple rip-first, then crosscut, then rerip operations—operations as practiced by softwood moulding and millwork industries. This program can be useful in studying yield of individual boards, yield of board grades and grade mixes, effects of alternative rip widths on yield, effects of defect scanning equipment, and effects of alternative cutting bills.

The program has been written in FORTRAN V and processed on the University of Wisconsin's UNIVAC 1110 computer. The complete program listing is included as an appendix.

Accession For	
NTIS GRA&I	☑
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A	

DTIC COPY INSPECTED 2

Table of Contents

United States
Department of
Agriculture

Forest Service

Forest
Products
Laboratory[1]

Research
Paper
FPL 412

September 1982

OPTYLD — A Multiple Rip-First Computer Program to Maximize Cutting Yields

By

Pamela J. Giese, Computer Programmer
and
Kent A. McDonald, Research Forest Products Technologist

Introduction

Improved utilization of lumber is critical both in battling increasing lumber costs and in conserving natural resources. Increasing yield is a basic step toward the goal of improved utilization. Higher yield means higher recovery of end product and decreased need of raw material input per unit output.

Measuring yield of a volume of lumber can be an elusive task. Most frequently, yield per volume is measured from unit input and estimated waste. This approach has led to the goal of reducing waste to increase yield indirectly. Within the softwood millwork industry, no yardstick exists to compare with this estimation of waste; aside from historical mill data, the only numerical value to judge by has been the impossible 0 percent waste. What is needed is information on the yield available within a given batch of lumber—what is the grade mix of the batch and what is the optimum yield per grade?

Program OPTYLD was designed to determine the maximum cutting yield of a board. This board by board yield has been the basis of a study to determine yields by board grade of 5/4 ponderosa pine for moulding and millwork industry application (*4*).[2] Additional applications of the OPTYLD program include studying effects of alternative cutting bills, effects of alternative rip widths, effects of various board defects, and recovery from defect scanning equipment.

Previous cutting yield simulation programs have based the cutting yield maximization on the surface area of a cutting or total surface area of the cutting within the board. (A discussion of OPTYLD vs. other cutting yield computer programs is given in Appendix C.) Difficulties arise with either of these approaches. Maximizing the surface area of a cutting (sawing to get the largest cutting available, then sawing the subsequent cuttings) may sacrifice the whole board to get one big cutting; maximizing the total surface area of cuttings may result in a solution that cuts up the board into a number of small cuttings. Program OPTYLD has been developed to relieve this dilemma. OPTYLD uses a value index table that assigns a value to each cutting dimension (a value can be considered to be simply a weighting factor) (table 1). Each column of table 1 corresponds to a cutting length range and each row to a cutting width range. For example, the value index of 830 is assigned to any cutting from 12.0 to 19.0 inches in length and from 2.25 to 3.25 inches in width.

OPTYLD maximizes total value of cuttings of the board. When the run-time option is to maximize total yield instead of total value, the value index table is set to all ones. Thus, the user controls the emphasis of cutting size selection. This control means that while the board is maximized as a whole, great flexibility exists in ranking the cutting yield solutions.

Description of Model

Computer program OPTYLD processes an unlimited number of boards one board at a time. It retains no memory of previous boards or their solutions. The program represents, as a binary matrix, a board measured as a rectangle superimposed on a Cartesian

[1] Maintained at Madison, Wis., in cooperation with the University of Wisconsin.

[2] Italicized numbers in parentheses refer to literature cited at the end of this report.

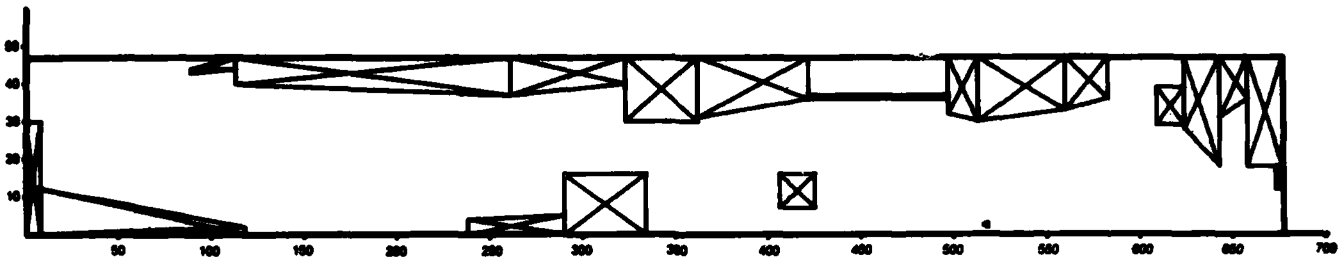


Figure 1. — Geometric representation of sample board No. 1306. Areas marked by an "X" are defects. All values are in quarter-inch units.

(M-149 797)

Table 1.—Value[1] indices used to calculate maximum value of sample board shown in figure 1.[2]

Cutting width in inches	Cutting length in inches								
	12	19	26	35	47	59	71	83	84
1.75	780	790	800	820	840	860	880	1000	1050
2.25	800	810	820	840	860	880	1000	1030	1150
3.25	810	830	840	860	880	910	1040	1070	1200
4.25	820	840	850	870	890	930	1080	1120	1250
4.75	825	845	855	875	895	950	1100	1145	1300

[1] These values are used for weighting and have no units.

[2] Each column corresponds to a cutting length range and each row to a cutting width range. For example, the value index of 830 is assigned to any cuttings 12.0 through 19.0 inches in length and 2.25 through 3.25 inches in width.

coordinate system with the lower left corner at the origin. Each cell within the binary matrix is set to 1 to represent a defect and 0 (zero) if not a defect. A board that has been so coded is plotted in figure 1. After the board matrix has been established, the process of locating the maximum ripping solution is begun. First, the board is ripped along its lower side to provide a clean, straight edge. The first valid rip width combination is generated. A valid rip width is one in which (1) the sum of the rip widths and their associated saw kerfs fits within the board, (2) the sum of the rip widths and saw kerfs when subtracted from the total width of the board gives a difference less than the width of the narrowest rip width,[3] and (3) the rip width combination has not been calculated. Using this rip width combination, the board is ripped and each rip is then crosscut for the longest possible cuttings. The values of the cuttings are totaled. The next valid rip combination is generated and the process is repeated. If this new total value is higher, it becomes the hign total value.

[3] Empirical testing has shown that rip width combinations that leave a salvage wider than the narrowest rip width seldom give the maximum yield and such a case occurs only when too few rip saws are specified at execution time. This constraint is very effective in reducing computer time.

After all valid rip combinations have been tried, the rip combination yielding the highest total value is again ripped and crosscut, but this time all areas not falling within a cutting are reripped for possible salvage cuttings. Salvage cuttings are not considered in the best rip combination decision. All rerip cuttings are variable length; the smallest length specified under the specified length option is considered the minimum rerip length.

A board area sent to be reripped is first scanned along its lower edge to determine if a minimum clear length for a cutting is present. If not, the top edge is scanned for a minimum clear length. If an acceptable length is found along one of the edges, the piece is reripped for all possible rerip widths. The value of the cutting found is compared to the previous highest valued rerip cutting. If the value of the new cutting is higher, it becomes the highest valued rerip cutting. This process is completed for both edges of the piece. Figure 2 and table 2 show possible rerip solutions.

Figure 3 and table 3 show an intermediate yielding rip combination and figure 4 and table 4 the maximum yielding rip width combination for the board pictured in figure 1. After this board has been ripped, crosscut, and reripped, the maximum value solution is printed (table 4) and the next board is read. (See Appendix E for listing of computer input and output used in processing the sample board (fig. 1)).

Description of Computer Program OPTYLD

OPTYLD is divided into 10 modules—the main program, 8 subroutines, and 1 function. Descriptions of these modules follow. Flowcharts showing the basic program and the more important subroutines are given in Appendix A (fig. 6-12).

Main Program

The main program (MAIN) is the input/output center of OPTYLD (Appendix A, fig. 7). At the beginning of a run, information concerning decision-making options, cutting lengths, rip widths, and rerip widths is read and set. A title page identifying the OPTYLD program options used is printed (table 5). The data for a board then are read and checked to see that the length,

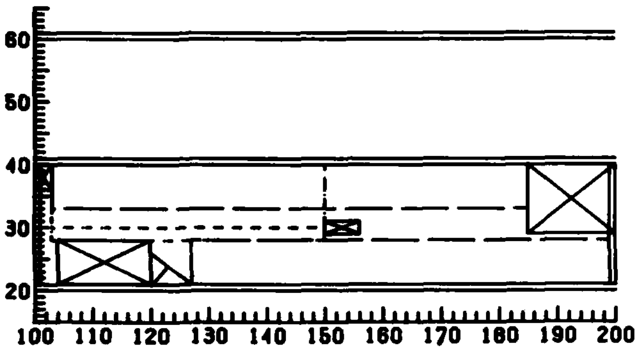


Figure 2.—A possible piece sent to be reripped. OPTYLD would choose the highest value cutting, crosscut and rerip to get the cutting, and then scan the remaining piece of lumber to be reripped. The four possible rerip cuttings and their coordinates are shown in table 2.
(M 149 798)

Table 2.—Four possible rerip cuttings and their coordinates for figure 2

Coordinates of piece to be reripped (see fig. 2): (101,21) to (199,40)
Dimensions of piece to be reripped: 24.5 × 4.75 in.
Possible rerip widths: 4.25 in., 3.50 in., 3.00 in., 2.50 in., 1.75 in.
Minimum length of a cutting: 9.0 in. or 36 quarter-inch units
In this piece, only 4 possible clear rerip cuttings would be found:

	Dimensions	Coordinates
(1)	18.0 in. × 1.75 in.	(127,21) to (199,28)
(2)	20.5 in. × 1.75 in.	(103,33) to (185,40)
(3)	11.75 in. × 2.50 in.	(103,30) to (150,40)
(4)	11.75 in. × 3.00 in.	(103,28) to (150,40)

width, and number of defects fit within the program array limits (see table 6 for program constraints). A board failing any check is discarded and the next board is read. A board passing all checks is transposed onto a two-dimensional binary grid where each grid represents a 1/4- inch by 1/4-inch area of the board and is encoded as a 1 if the grid represents a defective area of the board, or as a 0 if the grid represents a clear (nondefective) area. MAIN now calls subroutine COLECT, which coordinates the sawing of the board. When COLECT returns, MAIN calculates the final yield and value of the sum of all the cuttings of the maximum yielding solution. Output is then printed and/or filed as in table 4. All information on the finished individual board is discarded, and the next board is read.

Subroutine COLECT

Subroutine COLECT (Appendix A, fig. 8) acts as an organizer for the processing of the board. When called from MAIN, COLECT initializes both the number of cuttings and the board yield to zero, then calls INITAL (an entry of subroutine PERM2) to generate the first rip width combination. After INITAL returns, subroutine SAW is called to rip the board. When SAW returns with the yield information of the current rip combination, COLECT computes the percentage yield and the total value of all the cutting results of the current rip combination. COLECT then judges the current rip combination yield against the previous maximum rip combination yield. The mode of maximization is specified at run-time to be either the total area of cuttings or the total value of cuttings obtained by:

$$V = \frac{I}{1000} \times \frac{A}{(144)(16)}$$

where V = value of a cutting
I = value of this cutting size from value index table (when area of cuttings is to be maximized, I = 1000 for all cuttings)
A = area (length times width in quarter inches) of this cutting.

If the current yield exceeds the previous maximum yield, the current yield becomes the new maximum yield and PERM2 is then called to generate the next rip combination and that rip combination is computed. The process of calling PERM2, calling SAW, and comparing yields is repeated for all valid rip width combinations.

When all such combinations have been tried, the maximum yielding rip combination is computed again. This time areas not yielding clear cuttings are reripped in subroutine NOCHIP. Then COLECT returns. The maximum yield decision is based upon only full rip width cuttings and does not include rerip cuttings. This approach sets a bias against the additional cost of producing a rerip cutting as opposed to that of a full rip width cutting.

Subroutine PERM2

Subroutine PERM2 (Appendix A, fig. 9) generates rip width combinations to be sawn. PERM2 then tests whether the rip combination fits within the confines of the board, whether it has been generated before, and whether it leaves less than the maximum allowable salvage. If the combination fails any of these tests, the next rip combination is generated. If the combination passes all the tests, PERM2 returns to COLECT.

Subroutine SAW

Subroutine SAW (Appendix A, fig. 10) rips the board using the rip combination passed from PERM2 and searches for full width clear cuttings that meet cutting length requirements. When SAW is called from COLECT, it first checks to see if any rips of the new combination have been calculated before on this board. For example, if the present rip combination is 3.25 in., 3.25 in., 3.25 in., 4.25 in., and the previous rip combination of 3.25 in., 3.25 in., 3.25 in., 3.25 in. had already been calculated, only the last rip of 4.25 in. needs to be scanned for clear cuttings. The clear cutting yields of the previously calculated rips are retrieved by calling RETREV (an entry of subroutine VALSTO), as are the sawcut coordinates. The lower edge of the board is edged as specified by the run-time

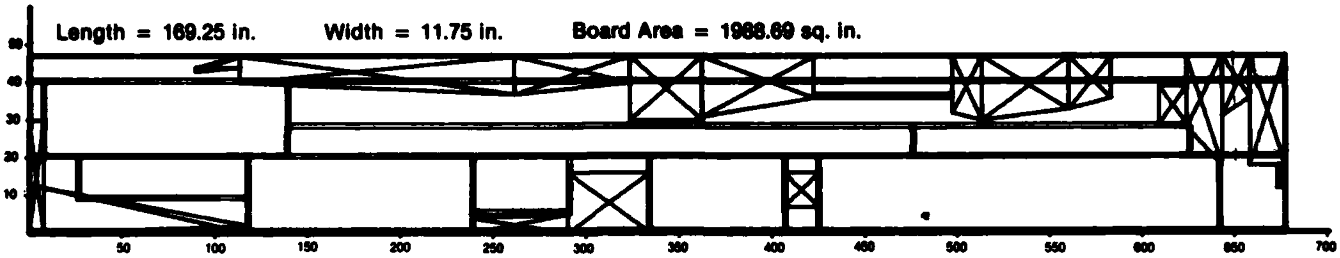


Figure 3. — Sample board No. 1306, rip width combination 4.75 inch, 4.75 inch. See table 3 for details.

(M 149 799)

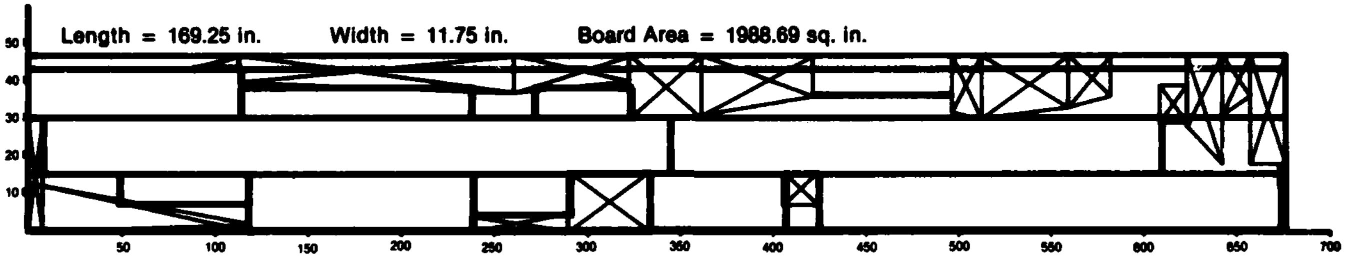


Figure 4. — Sample board No. 1306. Maximum value rip width combination. See table 4 for data.

(M 149 801)

Table 3. — Yield for the sample board shown in figure 3 when ripped with the rip width combination of 4.75 in. and 4.75 in.[1]

Dimension	Value	Coordinates			
		Lower X	Lower Y	Upper X	Upper Y
		FULL RIP CUTTINGS			
30.00 × 4.75	0.87	118	1	238	20
18.00 × 4.75	.50	334	1	406	20
54.00 × 4.75	1.69	425	1	641	20
32.50 × 4.75	.94	8	21	138	40

Total area = 638.87 sq. in.
Total value = 4.00
Percentage yield = 32.13 percent

Dimension	Value	Lower X	Lower Y	Upper X	Upper Y
		RERIP CUTTINGS			
22.50 × 2.50	0.33	27	10	117	20
12.75 × 3.50	.25	239	6	290	20
84.00 × 1.75	1.07	139	21	475	28
37.25 × 1.75	.38	476	21	625	28

Total area = 313.08 sq. in.
Total value = 2.03
Percentage yield = 15.74 percent

Total value of this rip combination (4.75 in., 4.75 in.) = 6.03
Total percentage yield of this rip combination = 47.87 percent

[1] This rip combination is not the maximum yield combination.

Table 4. — Yield for the sample board shown in figure 4 when ripped with the maximum value rip width combination for this board (3.50 in., 3.50 in., 3.00 in.)[1]

Dimension	Value	Coordinates			
		Lower X	Lower Y	Upper X	Upper Y
		FULL RIP CUTTINGS			
30.00 × 3.50	0.63	118	1	238	15
18.00 × 3.50	.37	334	1	406	15
62.00 × 3.50	1.63	425	1	673	15
84.00 × 3.50	2.55	8	16	344	30
66.00 × 3.50	1.73	345	16	609	30
28.25 × 3.00	.53	0	31	113	43

Total area = 994.75 sq. in.
Total value = 7.42
Percentage yield = 50.02 percent

Dimension	Value	Lower X	Lower Y	Upper X	Upper Y
		RERIP CUTTINGS			
17.00 × 1.75	0.16	49	8	117	15
12.75 × 2.50	.18	239	5	290	15
30.75 × 1.75	.31	114	31	237	38
12.75 × 1.75	.12	272	31	323	38

Total area = 137.75 sq. in.
Total value = 0.77
Percentage yield = 6.93 percent

Total value of this rip combination (3.50 in., 3.50 in., 3.00 in.) = 8.19
Total percentage yield of this rip combination = 56.95 percent

[1] See Appendix E for listing of computer input and output used in processing the sample board (fig. 1).

Table 5. — Options selected for computing maximum value of sample board shown in figure 1. Body of table is as it would appear in program output

Option	Selected values				
EDGING ALLOWANCE (IN.)	0.25				
CUTTING LENGTH OPTION	RANDOM				
RIP WIDTHS (IN.)	2.50	3.00	3.50	4.25	4.75
SALVAGE WIDTHS (IN.)	1.75	2.50	3.00	3.50	4.25
MINIMUM CUTTING LENGTH (IN.)	9.00				
MAXIMUM CUTTING LENGTH (IN.)	84.00				
RERIP LENGTHS	RANDOM				
VALUE MAXIMIZATION					

Table 6. — OPTYLD program constraints

Option constraints—Violation causes program to stop	
Maximum No.	
Ripsaws	8
Ripwidths	5
Rerip widths	5
Specified cutting lengths	10
Individual board constraints—Violation causes next board to be read	
Maximum No.	
Defects per board	100
Width of board	25 inches
Length of board	18 feet
Sawcuts per board	100
Number full rip width cuttings	50
Number rerip cuttings	30

option. For each remaining rip, SAW scans the full width of the rip for a clear allowable length; if such an area is found, SAW calls subroutine CROSS to place crosscuts. This scanning continues with the area of the rip immediately following the last crosscut. After all rips have been calculated, SAW returns to COLECT.

Subroutine CROSS

Subroutine CROSS (Appendix A, fig. 11) places a crosscut at the beginning of the last defect area (the end of the previously clear cutting). CROSS calculates the dimensions of the present cutting and calls subroutine VALSTO to store this information.

If the cutting length option has been set to saw specified lengths CROSS cuts out the longest specified length that will fit within the clear area, then calls VALSTO to store the information on size and location of the cutting.

When the maximum rip combination is being reprocessed, CROSS calls subroutine NOCHIP to calculate the rerip cuttings and then returns to SAW.

Subroutine NOCHIP

Subroutine NOCHIP (Appendix A, fig. 12) scans for usable cuttings within ripped areas that are not clear for the full width of the rip. Only cuttings that can be salvaged with one additional rip and only crosscuts across the full width of the rip are allowable. Beginning with the left edge of the piece, NOCHIP scans the lower edge, searching for a valid, clear length. If one is found, NOCHIP rerips the piece with the smallest rerip width. If this process yields a clear cutting, the next larger rerip width is tried. The process continues until all potential rerip widths are exhausted. When the process fails, the top edge is scanned for a clear length and processed again as above. The highest value rerip cutting is chosen and a crosscut is placed at the end of the cutting. NOCHIP then searches for more clear lengths along the edge. If it finds another, the reripping process is repeated (an example is shown in fig. 2 table 2). When both edges have been so examined for the length of the piece, NOCHIP returns.

Other Modules

Subroutines STORE AND STORE2, respectively, store the coordinates of sawcuts and rerip pieces. Function VALUE calculates the value of a cutting based upon (1) whether total area or total value of cuttings is to be maximized, (2) the square area of the cutting, and (3) the assigned value index of the cutting's dimension.

Input

Information input to run OPTYLD consists of two major types—the option deck and the board data deck. The option deck specifies alternatives in the decision-making faculties of the program. The board data deck contains information on individual boards such as the size and grade of the board and the size and locations of defects. Order of input must be an option deck followed by a board data deck. Several runs of alternating decks (i.e. option $deck_1$, board data $deck_1$, option $deck_2$, board data $deck_2$ etc.) may be submitted. The program uses logical means to determine the end of the option deck. The end of the board data deck is indicated to the program by an end of file mark or by the letters END on the card following the last board data card. The program stops when the character S is detected in column 11 of one of the first three option deck cards.

Option Deck

The option deck controls the alternatives that affect the decision-making and the cutting specifications under which the boards are processed. Table 7 shows the formatting of the cards making up the option deck The organization of this deck has been helpful in limiting costly mistakes in option misspecification.

Card 1. OPTYLD is designed to determine yields of either variable or specified length cuttings. A YES on this card signals the variable length option that means full rip width cuttings are bounded in length by defects, meet or exceed the minimum length (set on card 9), and

Table 7. — Option input summary

Card number	Card option	Alternatives/ values	Action of alternative/value	Card format
1	Length option— Are lengths to be unspecified?	YES	Cutting lengths are the longest possible length, referred to as random	10X, A1
		NO	Number of cutting lengths are specified on card 8 and lengths on card 13	10X, A1
2	Printing option—Is output to be suppressed?	YES	Printed output is suppressed (excluding error messages)	10X, A1
		NO	Printed output is not suppressed	10X, A1
3	Maximization option—Is a table of values to be used?	YES	Value of cutting is maximized (value table supplied)	10X, A1
		NO	Area of cuttings is maximized (value table not supplied)	10X, A1
4	Specifies depth of edging cut	$0 \leq I \leq 9$	Number of quarter-inch units to be edged from bottom of board	10X, I1
5	Specified maximum number of rips	$0 < I \leq 8$	The maximum number of rips to be taken	10X, I1
6	Specifies number of rip widths	$0 < I \leq 5$	The number of widths to be used in ripping, widths are specified on card 11	10X, I1
7	Specifies number of rerip widths	$0 < I \leq 5$	The number of widths to be used in reripping, widths are specified on card 12	10X, I1
8	Specifies number of cutting lengths	$0 < I \leq 10$	The number of specified cutting lengths, ignored if card 1 = YES. Lengths are specified on card 13	10X, I2
9	Minimum cutting length	Any positive real number	The minimum cutting length if card 1 = YES	10X, F6.2
10	Maximum cutting length	Any nonnegative	The maximum cutting length when card 1 = YES	10X, F6.2
11	Specified rip widths	One to five real numbers in ascending order	Rip to these widths	5F10.2
12	Specified rerip widths	One to five real numbers in ascending order	Rerip to these widths	5F10.2
13	Specified cutting lengths	One to ten real numbers in ascending order	Cutting lengths when card 1 = NO	10F8.2
	The following cards are present *only* when maximization is based upon total value (card 3 = YES). These cards create a value index table of 5 widths (rows) by 9 lengths (columns).			
14	Upper limits of cutting length groups	Nine integers (in ascending order)	Definition of the columns (length dimension of the value index table	9I8
15	Upper limit of width groups	Five integers in quarter-inch units (in ascending order)	Definition of the rows (width dimension) of the value index table	5I8
16-20	Values of a cutting within dimensions of a given cell	Nine integer values	The specified values are used to construct the value index table for a cutting whose dimensions are those of the cell	9I8

are less than or equal to the maximum allowable cutting length (set on card 10). A NO on this card signals the specified cutting length option and only those lengths specified on card 13 are allowed as full rip width cuttings. All rerip cuttings are variable length

Card 2.—At times it is desirable to suppress the printing of individual board data and secure all information from intermediate files. This is the purpose of card 2. A YES suppresses the printing of individual board information while a NO allows their printing. Error messages are printed regardless. (Possible error messages that can be printed are as follow: (1) the board's size or number of defects exceeds the array limits, (2) an array has overflowed, (3) the constraints of the program do not allow processing of a board, (4) an error was made within the option deck.)

Card 3.—OPTYLD allows two types of yield maximization—maximization on total value of cuttings or maximization on total area of cuttings. Choice of maximization of total value of cuttings, indicated by a YES on this card, allows the user to enter a value index table thereby assigning priority value to each cutting dimension. Cards 14-20 give specifications for the construction of the value index table. A NO on card 3 signals that maximization is to be based upon total surface area of cuttings.

Card 4.—A common industry practice is to straighten an edge of a board by removing a narrow strip or edging. While the best edging removed varies from board to board, it is a fixed amount in OPTYLD. This card specifies this edging width in quarter-inch units.

Card 5.—The maximum number of rip saws available to OPTYLD is specified on this card. This value is very important in determining the maximum solution. Ideally, the number of saws specified should be the same as the system being modeled. However, an exponential relationship exists between the number of rip saws and the number of combinations tried; as the number of saws increases, computing time skyrockets. Conversely, too few rip saws will not likely yield the "true" maximum combination in wide boards— the program may not even attempt to rip such boards. In choosing the number of rip saws, the user should carefully consider the range of board widths to be processed, the number of saws in the system being modeled, and any limits placed on the user by computing time. The program is currently set to handle a maximum of five rip widths and eight rip saws.

Card 6.—This card specifies the number of rip widths to be read from card 11.

Card 7.—This card specifies the number of rerip widths to be read from card 12.

Card 8.—This card indicates the number of specified cutting lengths that will be found on card 13. The data on this card are important only for specified lengths (Card 1 = NO). For variable cutting lengths (Card 1 = YES), the data on this card are ignored.

Card 9.—This card sets the minimum allowable lengtn of a cutting. The data on this card are important only when variable length cuttings are desired (Card 1 = YES).

Card 10.—This card sets the maximum allowable length of a cutting. The data on this card are important only when variable length cuttings are desired (Card 1 = YES). If the maximum allowable length is set to zero, the length of cuttings is unbounded.

Card 11.—The rip widths are specified on this card. Only the number of widths specified on Card 6 will be read. These widths are to be in inches (to the nearest quarter inch) and *must* be listed from narrowest to widest.

Card 12.—The rerip widths are specified on this card. Only the number of rerip widths specified on Card 7 will be read. These widths are to be in inches (to the nearest quarter inch) and *must* be listed from narrowest to widest.

Card 13.—The specified cutting lengths are given on this card. Only the number of lengths indicated on Card 8 will be read. These lengths are in inches (to the nearest quarter inch) and *must* be listed from shortest to longest. The data on this card are not applicable if Card 1 = YES.

The Value Index Table - Cards 14-20

When yield maximization is to be based upon total value of cuttings, a value index table must be supplied. However, these cards *must not* be included if maximization is based upon total area of cuttings.

Program OPTYLD is presently designed to read in and construct a 5 × 9 (widths × lengths) value index table (table 1).

Card 14.—This card specifies the length ranges (defines the columns) of the value index table. The maximum length of each range is an integer expressed as inches. For example, if a length class runs from 19.00 in. through 26.00 in., its column is represented as 26.

Card 15.—This card specifies the width categories (defines the rows) of the value index table. The maximum width of each width class is expressed as an integer number in quarter inch units. For example, if a width class runs from 2.25 in. through 3.25 in., its row is represented as 13.

Cards 16-20.—These five cards contain the actual body of the value index table. Each card contains the values of all columns for that particular row; Card 16 contains the values for all length classes for the first width class; Card 17 contains the values for all length classes for the second width class, and so on. All values are represented as integers.

Board Data Deck

The board data deck consists of information on each individual board. Cards are arranged such that all information of one board is read and processed before the next board is read. Each board within the board data deck contains three types of cards in the following sequence: one header card, one secondary card, and one or more defect data cards. The input for the sample board is shown in table 8.

The header card is the first card for each board and contains data on the grade, dimensions of the board, and the number of defects on the board. The dimension of the board is recorded as coordinates on a 1/4-in. by 1/4-in. grid system. Information on the header card is as follows:

Column	Information	Format
1 - 5	Board number	I5
6 - 8	Grade	A3
55 - 57	Number of defects on board	I3
61 - 62	Lower Y coordinate of board	I2
64 - 66	Lower X coordinate of board	I3
67 - 69	Upper Y coordinate of board	I3
70 - 73	Upper X coordinate of board	I4
79 - 80	Sequence number of card (= 1)	I2

The secondary card, always the second card for each board, contains information specific to the study, in this case ponderosa pine (*4*). The data contained on this card are for individual board information and play no role in program OPTYLD. The secondary card is formatted as such:

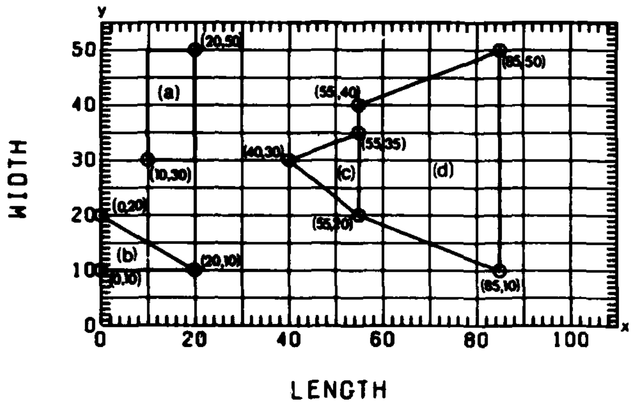
Column	Information	Format
1 - 5	Board number	I5
20 - 21	Surface measure of board	I2
79 - 80	Sequence number of card (= 2)	I2

The defect data card(s) come after these first two cards. Previous studies have documented procedures for coding defect data input as used in YIELD and MULRIP (*2,3*). However, formatting input used in OPTYLD is quite different. Basic defect input is processed as follow: (1) three defects are recorded per card; (2) a defect is treated as a quadrilateral; (3) seven entries are made for each defect—the type of defect, the left lower Y coordinate, the left upper Y coordinate, the left X coordinate, the right lower Y coordinate, the right upper Y coordinate, and the right X coordinate. Abbreviated methods of coding defects are allowed in the case of rectangles and triangles. These abbreviated coding methods and their corresponding quadrilateral representations are shown in figure 5. Any input recorded in an abbreviated method is translated into quadrilateral representation by OPTYLD. As in the first card, all coordinate values are in quarter-inch units. The following is the format for the defect cards:

Column	Information	Format
1 - 5	Board numbers	I5
7 - 8	Grade	A2
9 - 11	Defect type	A3
12 - 13	Left lower Y coordinate	I2
14 - 16	Left upper Y coordinate	I3
17 - 20	Left X coordinate	I4
21 - 23	Right lower Y coordinate	I3
24 - 26	Right upper Y coordinate	I3
27 - 30	Right X coordinate	I4
31 - 33	Defect type	A3
34 - 35	Left lower Y coordinate	I2
36 - 38	Left upper Y coordinate	I3
39 - 42	Left X coordinate	I4
43 - 45	Right lower Y coordinate	I3
46 - 48	Right upper Y coordinate	I3
49 - 52	Right X coordinate	I4
53 - 55	Defect type	A3
56 - 57	Left lower Y coordinate	I2
58 - 60	Left upper Y coordinate	I3
61 - 64	Left X coordinate	I4
65 - 67	Right lower Y coordinate	I3
68 - 70	Right upper Y coordinate	I3
71 - 74	Right X coordinate	I4
79 - 80	Sequence number of the card (≥ 3)	I2

Table 8. — Data input of sample board shown in figure 1

Card No.	Data input
1.	01306 XX SHOP 6/4 PONDEROSA PINE DEFECT TOTAL 19 00 000 47 677 01
2.	01306 + + 14 14 6/4 POND PINE 11 3 14 SAMPLE BOARD UNIT 013225 02
3.	01306 WN00 - 000 30 - 008 1 03
4.	01306 WN00 04 238 00 05 290 NK00 - 290 16 - 334 WN43 44 089 44 47 113 1 04
5.	01306 WN00 13 000 00 02 118 SP40 47 113 37 47 261 SP37 47 261 40 47 323 1 05
6.	01306 SP30 - 323 47 - 362 SP31 47 362 36 47 422 SP36 - 422 37 - 497 1 06
7.	01306 SP32 47 497 30 47 513 SP30 47 513 33 47 560 SP33 47 560 36 47 583 1 07
8.	01306 NK07 - 406 16 - 425 NK29 - 609 39 - 624 SP28 47 624 18 47 643 2 08
9.	01306 SP31 47 643 36 47 658 NK18 - 658 47 - 677 SC12 - 673 18 - 677 2 09



Defect (a) rectangle.

Abbreviated input description:	*30*	*-*	*010*	*50*	*-*	*020*
Quadrilateral representation:	*30*	*50*	*010*	*30*	*50*	*020*

Defect (b) "right-handed" triangle.

Abbreviated input description:	*10*	*20*	*000*	*10*	*-*	*020*
Quadrilateral representation:	*10*	*20*	*000*	*10*	*10*	*020*

Defect (c) "left-handed" triangle.

Abbreviated input description:	*30*	*-*	*040*	*20*	*35*	*055*
Quadrilateral representation:	*30*	*30*	*040*	*20*	*35*	*055*

Defect (d) quadrilateral.

Abbreviated input description:	None					
Quadrilateral representation:	*20*	*40*	*55*	*10*	*50*	*085*

Figure 5.—The four geometric representations of defects. The abbreviated method of coding a defect is shown above each quadrilateral representation of the defect. All values are in quarter-inch units.
(M 149 800)

Literature Cited

1. Hallock, Hiram.
   1980. Cutting yields fro· standard hardwood lumber grades when gang ripping. USDA For. Serv. Res. Pap. FPL 370, For. Prod. Lab., Madis..n, Wis.

2. Hallock, Hiram and Lynn Galiger.
   1971. Grading hardwood lumber by computer. USDA For. Serv. Res. Pap. FPL 157, For. Prod. Lab., Madison, Wis.

3. Lucas, Edwin L. and Leathern R. R. Catron.
   1973. A comprehensive defect data bank for No. 2 Common oak lumber. USDA For. Serv. Res. Pap. NE-262, Northeastern For. Exp. Stn., Upper Darby, Pa.

4. McDonald, Kent A., Richard O. Woodfin, and Pamela J. Giese.
   1981. 5/4 Ponderosa pine Shop grade cutting yields. USDA For. Serv. Res. Pap. FPL 394, For. Prod. Lab., Madison, Wis.

5. Schumann, David R.
   1972. Dimension yields from alder lumber. USDA For. Serv. Res. Pap. FPL 170, For. Prod. Lab., Madison, Wis.

6. Stern, Abigail R. and Kent A. McDonald.
   1978. Computer optimization of cutting yield from multiple ripped boards. USDA For. Serv. Res. Pap. FPL 318, For. Prod. Lab., Madison, Wis.

7. Wodzinski, Claudia and Eldona Hahm.
   1966. A computer program to determine yields of lumber. USDA For. Serv. FPL Unnumbered Publ., For. Prod. Lab., Madison, Wis.

Appendix A: Flowcharts for Program OPTYLD

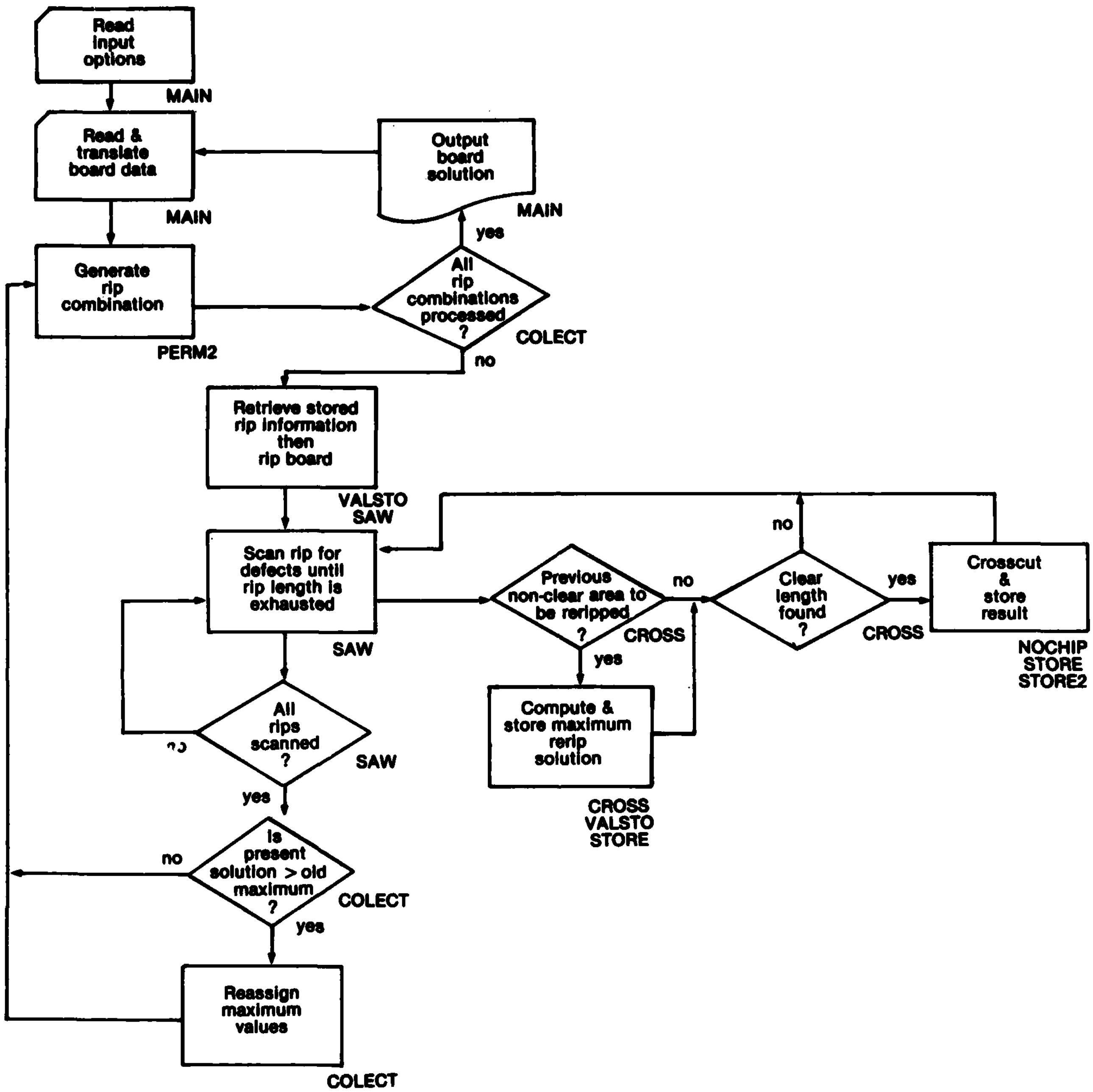


Figure 6. — General flowchart of program.

(M 151 572)

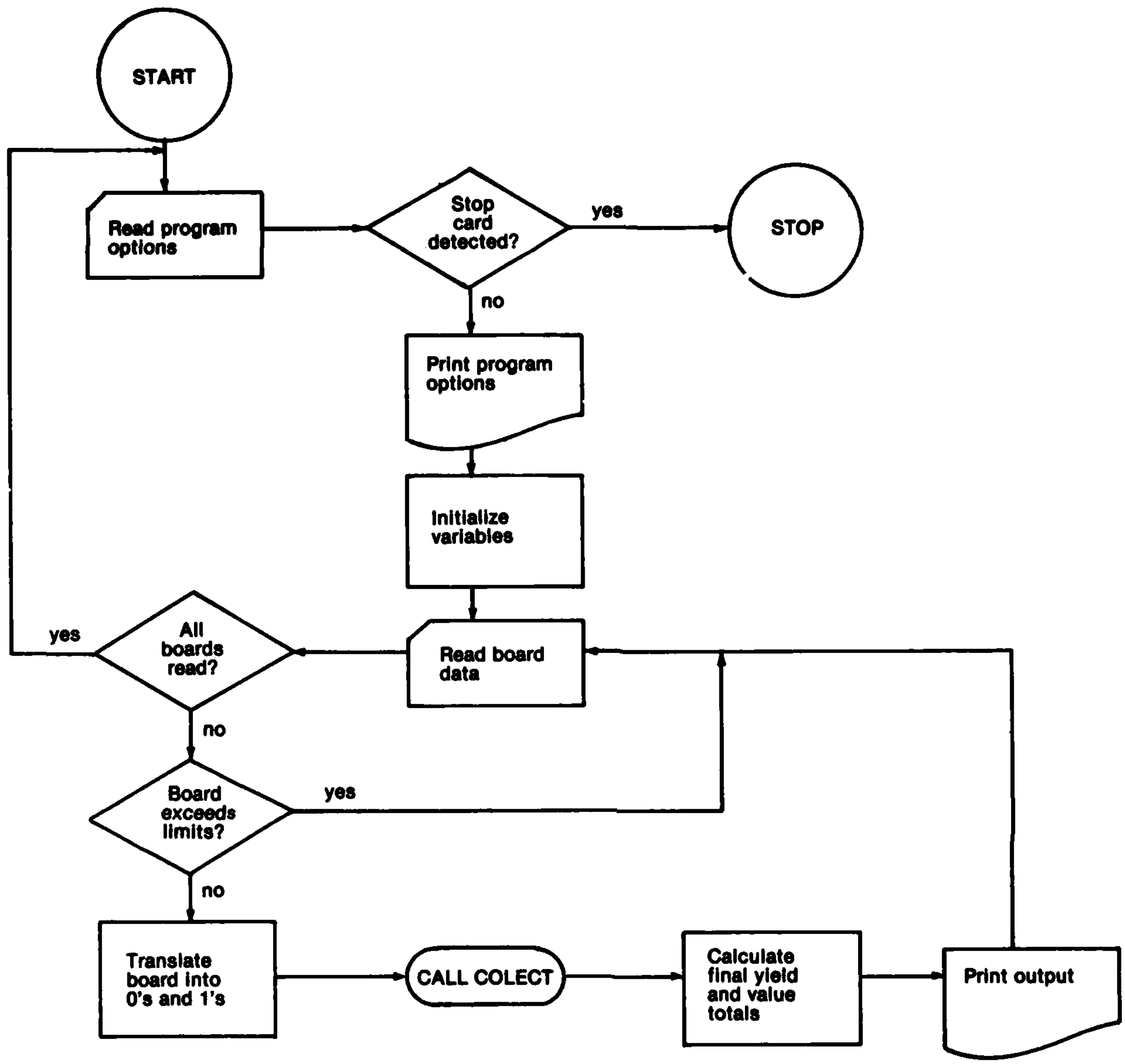


Figure 7. — Main program (MAIN).

(M 151 572)

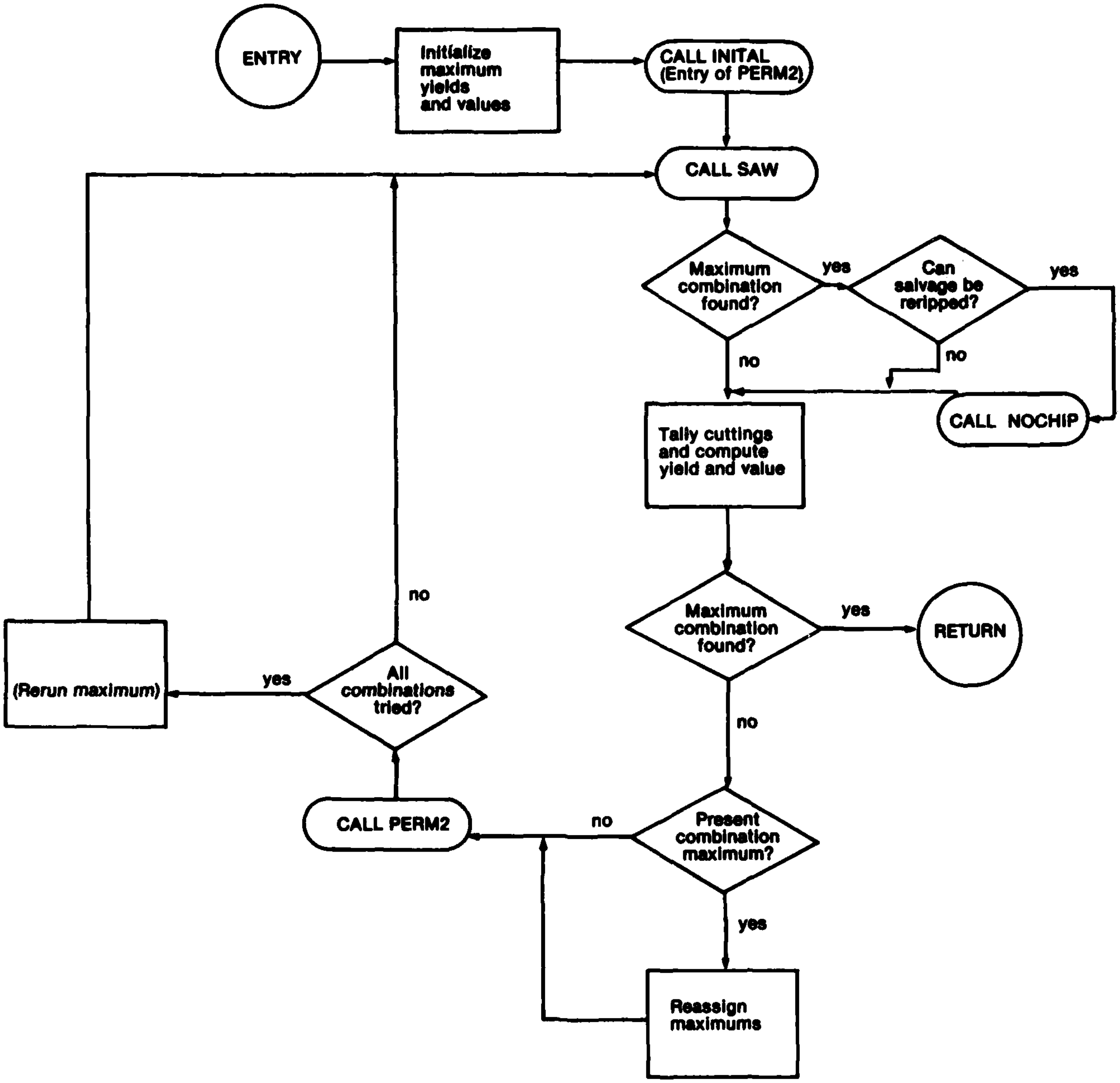


Figure 8. — Subroutine COLECT

(M 151 574)

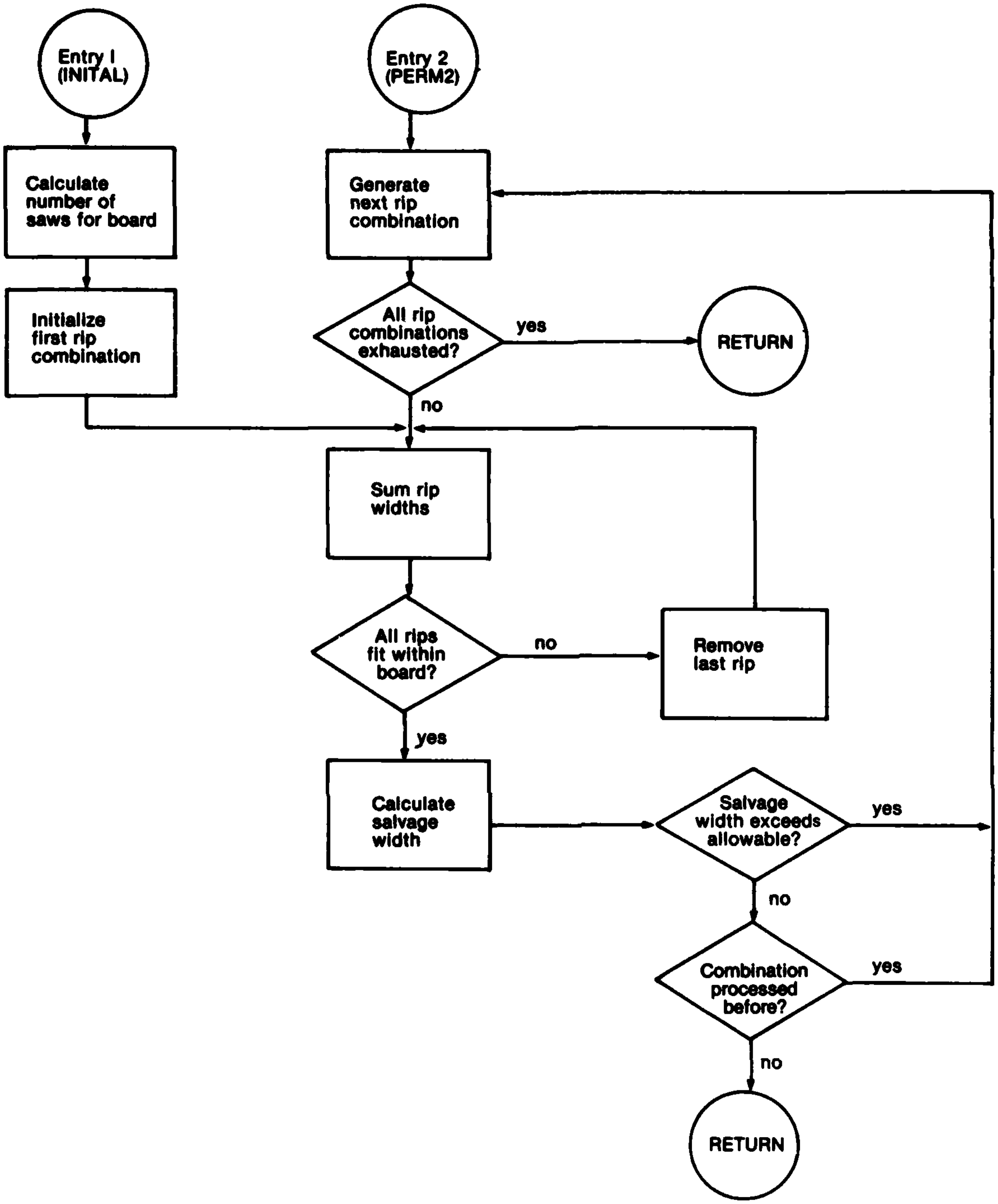


Figure 9. — Subroutine PERM2.

(M 151 575)

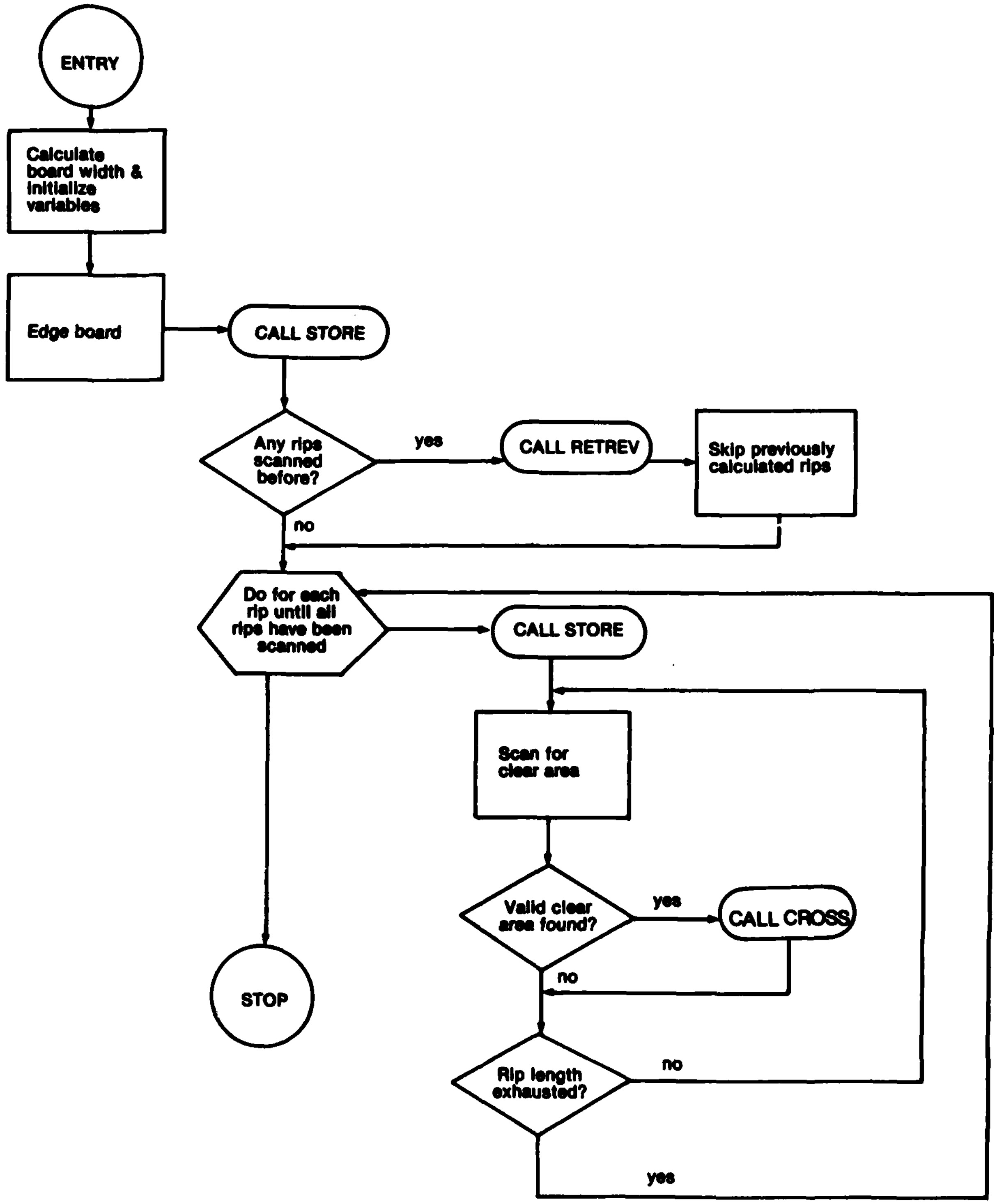


Figure 10. — Subroutine SAW.

(M151 576)

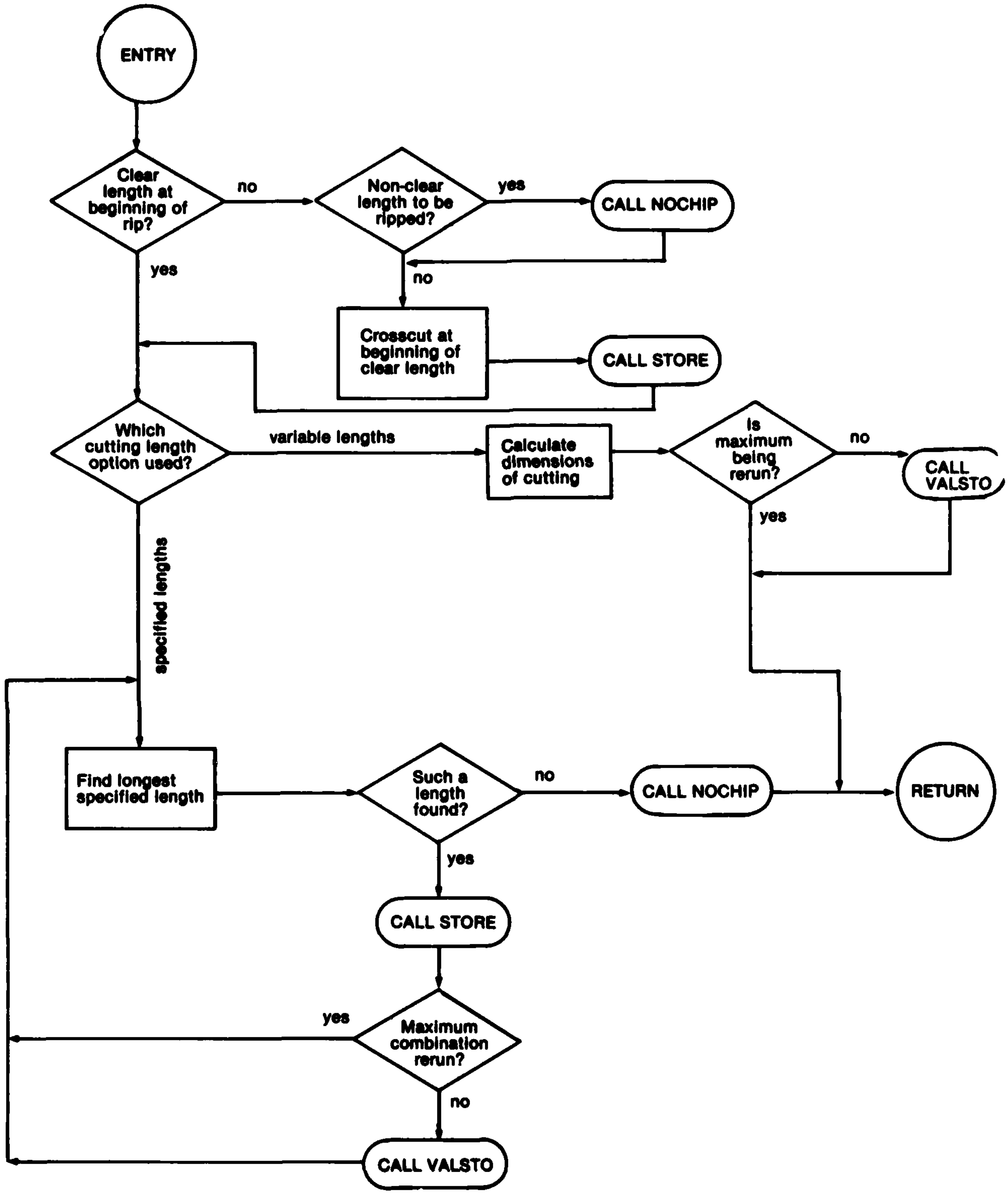


Figure 11. — Subroutine CROSS.

(M 151 577)

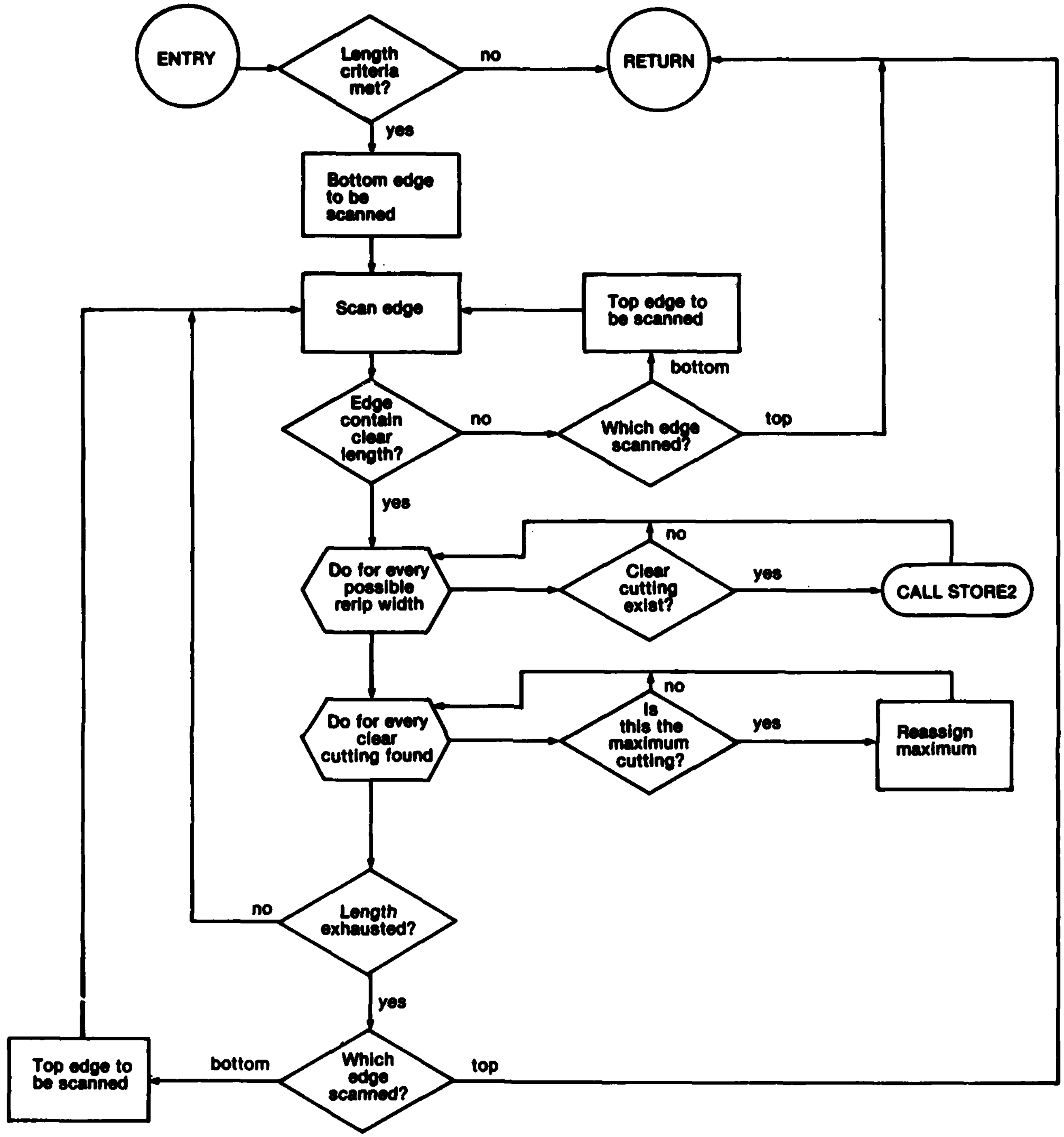


Figure 12. — Subroutine NOCHIP.

(M 151 578)

Appendix B: A Note on Use of Nonstandard FORTRAN

OPTYLD was written in FORTRAN V on a Sperry-UNIVAC 1110 and contains coding that may be specific to UNIVAC. The five literal arrays used in the program necessitate mentioning that the UNIVAC's FORTRAN V uses six characters per word. A reread of a record is accomplished by setting the logical unit device to zero. This is done in subroutine MAIN where the program is searching for the next board after discovering a board exceeding the allowable number of defects. Logical input devices used are 5 (card input), 6 (line printer output), 9 (disk file), and 11 (disk file).

In translating a board into 0's and 1's, a field function is used to access bits of a 36-bit word. The function has the form

FLD(k,i,j)

where k = number of the bit, i = width of the value in bits (always equal to 1), j = integer variable of the board array.

This function is necessary to pack the board into a reasonable array size. Any packing of the array necessitates computer specific instructions.

Multiple return addresses used in calls to PERM2 from subroutine COLECT may also cause problems on some machines.

Because of the organization of the data used in developing this program, tests were used to check for the presence of a -0. This will cause problems in two's-complement machines and will necessitate a slight change in the data and/or program if the same data format is to be used.

Appendix C: OPTYLD Vs. Similar Computer Programs

In discussing the mathematical model behind OPTYLD, it is helpful to distinguish between OPTYLD and similar models. Program YIELD (*7*) is one of the first cutting yield computer simulation programs. YIELD was designed to simulate practices of the hardwood furniture manufacturing industry and emphasizes crosscut-first decisions. In addition, YIELD does not maximize total value or yield of cuttings, as does OPTYLD. Yield nomograms such as those derived from YIELD (*5*) have also been derived from program MULRIP (*1*) (Abigail R. Stern, unpublished computer program, 1978, FPL), a precursor in the development of OPTYLD. An important distinction between programs YIELD and OPTYLD is in the process of locating clear areas. The term clear areas shall be used in this paper to mean an area within the board whih is free of defect. YIELD defines a clear area by searching an array of defect areas and previously defined cutting areas and testing for defects. If no defects or previously located cuttings are found, the area is considered clear. OPTYLD defines a clear area by scanning the length and breadth of the area in question, searching for a defect. If no defects are found, the area is clear. The process employed by OPTYLD is a better model for automatic scanning since it simulates the actual action of an automatic defect scanner.

In YIELD, cutting widths and cutting lengths must be specified. OPTYLD requires that cutting widths be specified, but lengths needn't be specified. If lengths are not specified, only the minimum allowable length is required. Delineating a maximum length is optional.

Another major difference between OPTYLD and YIELD is in the influencing of maximization decisions. OPTYLD maximizes the total yield of cuttings within the board. When maximization of yield is to be based upon total value of cuttings rather than total area of cuttings, OPTYLD uses a value index table to calculate the value of each cutting. YIELD maximizes the largest clear area. This gives a solution which may sacrifice the whole board to get out a single large cutting.

Computer program RIPYLD (*6*) was the first of the multiple-rip computer models to actually simulate the scanning process. RIPYLD received input from an automatic scanner where data were represented in binary form. Using algorithms similar to those in OPTYLD, RIPYLD found the highest yielding rip width combination by ripping and crosscutting to obtain the largest total area of cuttings; no reripping was employed.

Appendix D: Listing of Computer Program OPTYLD

Cross Reference Used

Following the listing of each subroutine or function, a cross reference is given. FORTRAN statement numbers are listed first. Following each statement number is the line number of the statement(s) that reference the statement number. An asterisk indicates that this line lumber defines the statement number. The listing of the variables is similar to the listing of the statement numbers. Variables are listed in alphabetic order and the line number(s) referencing the variables listed after it. An asterisk indicates that the variable is defined or redefined at that line number.

```
  1 C *** OPTYLD - RIP FIRST. CROSSCUT. AND SALVAGE  -  VERSION P1:981    OPTYLD
  2 C *** PROCESSES DATA FORMATTED AS IN PONDEROSA PINE MOULDING           OPTYLD
  3 C *** AND MILLWORK YIELD STUDY OF 1978                                 OPTYLD
  4 C *** EMPHASIS ON FINGERJOINTING                                       OPTYLD
  5       IMPLICIT INTEGER(A-Z)                                            OPTYLD
  6       REAL A.AREA.GR1.GR2.GR3.GRID.KERF.L.LSLOPE.LTEMP.RISE1.RISE2.    OPTYLD
  7      1     RTEMP.SAWMAX.SAWMIN.TEMWID(10).URTIMS.URTIMG.USLOPE.V.      OPTYLD
  8      2     VALUE.VTEMP.W.WORTH.WTEMP.YIELD.YLD.YTEMP1.YTEMP2           OPTYLD
  9       LOGICAL RANDOM.PRTOUT.DOLMAX                                     OPTYLD
 10       DIMENSION OPTION(3).TEXT(3).FOR3(7).FOR4(2).CLASS(100)           OPTYLD
 11      1     .DLX(100).DLY1(100).DUY1(100).DUX(100).DLY2(100).DUY2(100)  OPTYLD
 12       COMMON /MD/ MAX.YIELD(2).NP.AREA.DOLMAX.SALMIN                   OPTYLD
 13       COMMON /MP/ NWIDTH.NSAW.KERF                                     OPTYLD
 14       COMMON /MD/NSEC.RERIP(30.2).SAL(30.4)                            OPTYLD
 15       COMMON /MOSP/ GRID.NARROW                                        OPTYLD
 16       COMMON /MSD/BOARD(100.24).MAXLEN.BDNUM.MINLEN                    OPTYLD
 17       COMMON /MHOV/ SAWCUT(100.4).AVAIL                                OPTYLD
 18       COMMON /MOCV/ NPIECE.PIECE(50.2)                                 OPTYLD
 19       COMMON /MPSC/ WIDTH(5)                                           OPTYLD
 20       COMMON /MC/ LENGTH(10).NLEN                                      OPTYLD
 21       COMMON /MD2/NREWID.REWID(5).SALVAG(15.4.5.2)                     OPTYLD
 22       COMMON /MCS/RANDOM                                               OPTYLD
 23       COMMON /MW/ VALUES(10.5).LVALUE(10).WVALUE(5)                    OPTYLD
 24       COMMON /MOD/ BDLX.BDUX.BDLY.BDUY                                 OPTYLD
 25       DATA UNIT/5/                                                     OPTYLD
 26       DATA UNIT1/9/                                                    OPTYLD
 27       DATA UNIT2/5/                                                    OPTYLD
 28       DATA UNIT3/11/                                                   OPTYLD
 29       DATA VALUES/50*1/                                                OPTYLD
 30       DATA TEXT/18HSPECIFIED   RANDOM/                                 OPTYLD
 31       DATA FOR3/42H(1H0.10X'CUTTING LENGTH OPTION'.18X.                OPTYLD
 32       DATA FOR4/12H A6)  2A6)  /                                       OPTYLD
 33       DATA LVALUE/10*100/                                              OPTYLD
 34       DATA WVALUE/5*100/                                               OPTYLD
 35 C *** READ AND SET OPTIONS                                             OPTYLD
 36 C                                                                      OPTYLD
 37       FIRST=1                                                          OPTYLD
 38   100 DO 105 I=1.3                                                     OPTYLD
 39         READ(UNIT2.1)TEMP                                              OPTYLD
 40         IF(TEMP.EQ.'S')STOP                                            OPTYLD
 41         IF(TEMP.EQ.'Y')OPTION(I)=1                                     OPTYLD
 42         IF(TEMP.EQ.'N')OPTION(I)=0                                     OPTYLD
 43   105 CONTINUE                                                         OPTYLD
 44       READ(UNIT2.2)NARROW                                              OPTYLD
 45       GRID=.25                                                         OPTYLD
 46       KERF=.25                                                         OPTYLD
 47       RTEMP=FLOAT(NARROW)*GRID                                         OPTYLD
 48       READ(UNIT2.2)NSAW                                                OPTYLD
 49       READ(UNIT2.2)NWIDTH                                              OPTYLD
 50       READ(UNIT2.2)NREWID                                              OPTYLD
 51       READ(UNIT2.3)NLEN                                                OPTYLD
 52       READ(UNIT2.4)SAWMIN                                              OPTYLD
 53       READ(UNIT2.4)SAWMAX                                              OPTYLD
 54       PRTOUT=.TRUE.                                                    OPTYLD
 55       DOLMAX=.FALSE.                                                   OPTYLD
 56       RANDOM=.FALSE.                                                   OPTYLD
 57       IF(OPTION(1).EQ.1) RANDOM=.TRUE.                                 OPTYLD
 58       IF(OPTION(2).EQ.1)PRTOUT=.FALSE.                                 OPTYLD
 59       IF(OPTION(3).EQ.1)DOLMAX=.TRUE.                                  OPTYLD
 60       IF((NSAW.GT.8.OR.NWIDTH.GT.5).OR.NLEN.GT.10) GO TO 295           OPTYLD
 61       IF(NREWID.GT.5)GO TO 295                                         OPTYLD
 62       READ(UNIT2.5)(TEMWID(I).I=1.NWIDTH)                              OPTYLD
 63       WRITE(6.6)RTEMP                                                  OPTYLD
 64       FOR3(7)=FOR4(2)                                                  OPTYLD
 65       IF(RANDOM) FOR3(7)=FOR4(1)                                       OPTYLD
 66       IF(.NOT.RANDOM)WRITE(6.FOR3) TEXT(1).TEXT(2)                     OPTYLD
 67       IF(RANDOM)WRITE(6.FOR3) TEXT(3)                                  OPTYLD
 68       WRITE(6.7) (TEMWID(I).I=1.NWIDTH)                                OPTYLD
 69       DO 110 I=1.NWIDTH                                                OPTYLD
 70         WIDTH(I)=IFIX(TEMWID(I)/GRID)                                  OPTYLD
 71   110 CONTINUE                                                         OPTYLD
 72       READ(UNIT2.5)(TEMWID(I).I=1.NREWID)                              OPTYLD
 73       WRITE(6.8)(TEMWID(I).I=1.NREWID)                                 OPTYLD
 74       DO 115 I=1.NREWID                                                OPTYLD
 75         REWID(I)=IFIX(TEMWID(I)/GRID)                                  OPTYLD
 76   115 CONTINUE                                                         OPTYLD
 77       SALMIN=REWID(1)                                                  OPTYLD
 78       READ(UNIT2.9) (TEMWID(I).I=1.NLEN)                               OPTYLD
 79       IF(.NOT.RANDOM)SAWMIN=TEMWID(1)                                  OPTYLD
 80       IF(.NOT.RANDOM)SAWMAX=TEMWID(NLEN)                               OPTYLD
 81       MINLEN=IFIX((SAWMIN*4.)+.005)                                    OPTYLD
 82       MAXLEN=IFIX((SAWMAX*4.)+.005)+1                                  OPTYLD
 83       IF(RANDOM)WRITE(6.10) SAWMIN                                     OPTYLD
 84       IF(RANDOM.AND.SAWMAX.GT.0)WRITE(6.11)SAWMAX                      OPTYLD
 85       IF(.NOT.RANDOM)WRITE(6.12) (TEMWID(I).I=1.NLEN)                  OPTYLD
 86       WRITE(6.13)                                                      OPTYLD
 87       DO 120 I=1.NLEN                                                  OPTYLD
 88         LENGTH(I)=INT(TEMWID(I)/GRID)                                  OPTYLD
 89   120 CONTINUE                                                         OPTYLD
 90       IF(DOLMAX)WRITE(6.14)                                            OPTYLD
 91       IF(.NOT.DOLMAX)WRITE(6.15)                                       OPTYLD
 92       IF(.NOT. DOLMAX)GO TO 135                                        OPTYLD
 93       READ(UNIT2.16)(LVALUE(I).I=1.9)                                  OPTYLD
 94       READ(UNIT2.17) (WVALUE(I).I=1.5)                                 OPTYLD
 95       WRITE(6.18)(LVALUE(I).I=1.9)                                     OPTYLD
 96       DO 125 J=1.5                                                     OPTYLD
 97         READ(UNIT2.16)(VALUES(I.J).I=1.9)                              OPTYLD
 98         WTEMP=WVALUE(J)*GRID                                           OPTYLD
 99         WRITE(6.20)WTEMP.(VALUES(I.J).I=1.9)                           OPTYLD
100   125 CONTINUE                                                         OPTYLD
101       WRITE(6.21)                                                      OPTYLD
102 C                                                                      OPTYLD
103 C *** READ BOARD AND DEFECT INFORMATION                                 OPTYLD
104 C *** FUNCTIONS URTIMG AND URTIMS ARE UNIVAC SYSTEM FUNCTIONS TO SET    OPTYLD
105 C *** THE CLOCK                                                        OPTYLD
106   135 IF(FIRST.GT.1)CALL URTIMG('END')                                 OPTYLD
107       CALL URTIMS(0.0)                                                 OPTYLD
108       READ(UNIT.22.END=100) BDNUM.GR1.GR2.GR3.ND.BDLY.BDLX.BDUY.BDUX   OPTYLD
109       FIRST=2                                                          OPTYLD
110       UNIT=5                                                           OPTYLD
111       NSEC=0                                                           OPTYLD
```

```
      IF(BDNUM.EQ.'END   ')GO TO 300                                   OPTYLD
      DO 145 I=1,100                                                   OPTYLD
        DO 140 J=1,24                                                  OPTYLD
          BOARD(I,J)=0                                                 OPTYLD
  140   CONTINUE                                                       OPTYLD
  145 CONTINUE                                                         OPTYLD
      IF(PRTOUT)WRITE(6,23) GR1,GR2,GR3,BDNUM                          OPTYLD
      READ(UNIT,24)BDNUM,SM,NUNIT                                      OPTYLD
      NP=BDUX-BDLX                                                     OPTYLD
      AREA=NP*(BDUY-BDLY)*(GRID**2)                                    OPTYLD
      L=NP*GRID                                                        OPTYLD
      W=(BDUY-BDLY)*GRID                                               OPTYLD
      IF(PRTOUT)WRITE(6,25) L,W                                        OPTYLD
      I1=0                                                             OPTYLD
C *** TEST THAT BOARD FITS WITHIN ARRARY LIMITS                          OPTYLD
      IF(ND.LE.100) GO TO 160                                          OPTYLD
      WRITE(6,26)BDNUM                                                 OPTYLD
      GO TO 170                                                        OPTYLD
  160 IF(BDUY.LT.100) GO TO 165                                        OPTYLD
      WRITE(6,27)BDNUM                                                 OPTYLD
      GO TO 170                                                        OPTYLD
  165 IF(BDUX.LT.864) GO TO 175                                        OPTYLD
      WRITE(6,28)BDNUM                                                 OPTYLD
  170 READ(5,29)NSEQ                                                   OPTYLD
      IF(NSEQ.NE.1)GO TO 170                                           OPTYLD
      UNIT=0                                                           OPTYLD
      GO TO 135                                                        OPTYLD
  175 IF(ND.EQ.0)GO TO 210                                             OPTYLD
      DO 205 I2=1,ND                                                   OPTYLD
        IF(I2.LE.I1)GO TO 205                                          OPTYLD
        I1=I2+2                                                        OPTYLD
        READ(UNIT,30) (CLASS(I),DLY1(I), DUY1(I),DLX(I),DLY2(I),       OPTYLD
     1  DUY2(I),DUX(I),I=I2,I1)                                        OPTYLD
        J2=I2                                                          OPTYLD
        J1=I1                                                          OPTYLD
        DO 180 I=J2,J1                                                 OPTYLD
          IF(CLASS(I).NE.'   ')GO TO 180                               OPTYLD
          I1=I1-1                                                      OPTYLD
  180   CONTINUE                                                       OPTYLD
        DO 200 I=I2,I1                                                 OPTYLD
          IF(I.GT.ND)GO TO 200                                         OPTYLD
          IF(DUY1(I).EQ.-0)GO TO 185                                   OPTYLD
          IF(DUY2(I).NE.-0)GO TO 200                                   OPTYLD
C *** PLACE DEFECT COORDINATES IN CORRECT ORDER                          OPTYLD
C *** DEFECT IS LEFTHANDED TRIANGLE                                      OPTYLD
          DUY2(I)=DLY2(I)                                              OPTYLD
          GO TO 200                                                    OPTYLD
  185     IF(DUY2(I).EU.-0)GO TO 190                                   OPTYLD
C *** DEFECT IS RIGHTHANDED TRIANGLE                                     OPTYLD
          DUY1(I)=DLY1(I)                                              OPTYLD
          GO TO 200                                                    OPTYLD
C *** DEFECT IS PECTANGLE                                                OPTYLD
  190     DUY1(I)=DLY2(I)                                              OPTYLD
          DUY2(I)=DLY2(I)                                              OPTYLD
          DLY2(I)=DLY1(I)                                              OPTYLD
C *** DEFECT IS A TRAPAZOID                                              OPTYLD
  200   CONTINUE                                                       OPTYLD
  205 CONTINUF                                                         OPTYLD
C *** STORE DEFECT DATA AS A GRID OF 0'S AND 1'S FOR THE ENTIRE BOARD.   OPTYLD
C                                                                      OPTYLD
  210 IF(ND.EQ.0) GO TO 240                                            OPTYLD
      DO 235 K=1,ND                                                    OPTYLD
        Y1=DLY1(K)+1                                                   OPTYLD
        Y2=DUY1(K)                                                     OPTYLD
        X1=DLX(K)+1                                                    OPTYLD
        X2=DUX(K)                                                      OPTYLD
        YTEMP1=FLOAT(Y1)                                               OPTYLD
        YTEMP2=FLOAT(Y2)                                               OPTYLD
        RISE1=DUY2(K)-DUY1(K)                                          OPTYLD
        RISE2=DLY2(K)-DLY1(K)                                          OPTYLD
        RUN=DUX(K)-DLX(K)                                              OPTYLD
        USLOPE=RISE1/RUN                                               OPTYLD
        LSLOPE=RISE2/RUN                                               OPTYLD
        DO 230 I=X1,X2                                                 OPTYLD
          K2=I/36                                                      OPTYLD
          BIT=IABS(I-K2*36)                                            OPTYLD
          K2=K2+1                                                      OPTYLD
          IF(Y2.EQ.0)GO TO 225                                         OPTYLD
          DO 220 J=Y1,Y2                                               OPTYLD
            K1=J                                                       OPTYLD
            FLD(BIT,1,BOARD(K1,K2))=1                                  OPTYLD
  220     CONTINUE                                                     OPTYLD
  225     YTEMP1=YTEMP1+LSLOPE                                         OPTYLD
          YTEMP2=YTEMP2+USLOPE                                         OPTYLD
          Y1=IFIX(YTEMP1+.5)                                           OPTYLD
          Y2=IFIX(YTEMP2+.5)                                           OPTYLD
          IF(Y1.LT.0)Y1=0                                              OPTYLD
          IF(Y2.LT.0)Y2=0                                              OPTYLD
  230   CONTINUE                                                       OPTYLD
  235 CONTINUE                                                         OPTYLD
C                                                                      OPTYLD
C *** PREPARE RESULTS FOR OUTPUT                                         OPTYLD
  240 CALL COLECT                                                      OPTYLD
      IF(NP.LE.-1)WRITE(6,31)BDNUM                                     OPTYLD
      IF(NP.LE.-1)GO TO 135                                            OPTYLD
      A=0.                                                             OPTYLD
      V=0.                                                             OPTYLD
      YLD=0.                                                           OPTYLD
      WORTH=0.                                                         OPTYLD
      IF(NPIECE.EQ.0) GO TO 260                                        OPTYLD
      IF(PRTOUT.AND.DOLMAX)WRITE(6,32)                                 OPTYLD
      IF(PRTOUT.AND..NOT.DOLMAX)WRITE(6,33)                            OPTYLD
      DO 250 I=1,NPIECE                                                OPTYLD
        VTEMP=VALUE(PIECE(I,1),PIECE(I,2))                             OPTYLD
        LTEMP=FLOAT(PIECE(I,1))*GRID                                   OPTYLD
        WTEMP=FLOAT(PIECE(I,2))*GRID                                   OPTYLD
        IF(PRTOUT)WRITE(6,34)LTEMP,WTEMP,VTEMP                         OPTYLD
        A=A+LTEMP*WTEMP                                                OPTYLD
        V=V+VTEMP                                                      OPTYLD
  250 CONTINUE                                                         OPTYLD
      WORTH=WORTH+V                                                    OPTYLD
      IF(PRTOUT.AND.DOLMAX)WRITE(6,35) A,V                             OPTYLD
      IF(PRTOUT.AND..NOT.DOLMAX)WRITE(6,36) A,V                        OPTYLD
      YLD=(A/AREA)*100.                                                OPTYLD
      IF(PRTOUT)WRITE(6,37) YLD                                        OPTYLD
  260 A=0.                                                             OPTYLD
      V=0.                                                             OPTYLD
      YLD=0.                                                           OPTYLD
      IF(NSEC.EQ.0) GO TO 270                                          OPTYLD
      IF(PRTOUT.AND.DOLMAX)WRITE(6,38)                                 OPTYLD
      IF(PRTOUT.AND..NOT.DOLMAX)WRITE(6,39)                            OPTYLD
      DO 265 I=1,NSEC                                                  OPTYLD
        VTEMP=VALUE(RERIP(I,1),RERIP(I,2))                             OPTYLD
        LTEMP=FLOAT(RERIP(I,1))*GRID                                   OPTYLD
        WTEMP=FLOAT(RERIP(I,2))*GRID                                   OPTYLD
        IF(PRTOUT)WRITE(6,34) LTEMP,WTEMP,VTEMP                        OPTYLD
        A=A+LTEMP*WTEMP                                                OPTYLD
        V=V+VTEMP                                                      OPTYLD
  265 CONTINUE                                                         OPTYLD
      WORTH=WORTH+V                                                    OPTYLD
      IF(PRTOUT)WRITE(6,35) A,V                                        OPTYLD
      YLD=(A/AREA)*100.                                                OPTYLD
      IF(PRTOUT)WRITE(6,37) YLD                                        OPTYLD
  270 IF(PRTOUT)WRITE(6,40) AREA                                       OPTYLD
      YIELD(MAX)=YIELD(MAX)+YLD                                        OPTYLD
      IF(PRTOUT)WRITE(6,41)YIELD(MAX)                                  OPTYLD
      IF(PRTOUT.AND.DOLMAX)WRITE(6,42)WORTH                            OPTYLD
      IF(PRTOUT.AND..NOT.DOLMAX)WRITE(6,43)WORTH                       OPTYLD
      WRITE(UNIT1,-)BDNUM,SM,NPIECE,NSEC,AREA,YIELD(MAX),WORTH         OPTYLD
      WRITE(UNIT1,-)BDNUM,((PIECE(I,J),J=1,2),I=1,NPIECE),             OPTYLD
     1             ((RERIP(I,J),J=1,2),I=1,NSEC)                       OPTYLD
      IF(.NOT.PRTOUT)GO TO 135                                         OPTYLD
```

```
      IF(AVAIL.EQ.0) GO TO 205                                            OPTYLD
       WRITE(6.44)                                                        OPTYLD
      T=AVAIL-1                                                           OPTYLD
      WRITE(UNIT3.45)BDNUM.T                                              OPTYLD
      DO 200 I=1.T                                                        OPTYLD
        WRITE(6.46) (SAWCUT(I.J).J=1.4)                                   OPTYLD
        WRITE(UNIT3.46)(SAWCUT(I.J).J=1.4)                                OPTYLD
  200 CONTINUE                                                            OPTYLD
  205 IF(NSEC.EQ.0) GO TO 135                                             OPTYLD
      WRITE(6.47)                                                         OPTYLD
      DO 290 I=1.NSEC                                                     OPTYLD
      WRITE(6.46) (SAL(I.J).J=1.4)                                        OPTYLD
  290 CONTINUE                                                            OPTYLD
      GO TO 135                                                           OPTYLD
  295 WRITE(6.40)                                                         OPTYLD
  300 STOP                                                                OPTYLD
    1 FORMAT(10X.A1)                                                      OPTYLD
    2 FORMAT(10X.I1)                                                      OPTYLD
    3 FORMAT(10X.I2)                                                      OPTYLD
    4 FORMAT(10X.F6.2)                                                    OPTYLD
    5 FORMAT(5F10.2)                                                      OPTYLD
    6 FORMAT(1H1///38X.'OPTYLD'///10X'EDGING ALLOWANCE (IN.)'.17X.F5.2)   OPTYLD
    7 FORMAT(1H0.10X'RIP WIDTHS (IN.)'.23X.5F7.2)                         OPTYLD
    8 FORMAT(1H0.10X.'SALVAGE WIDTHS (IN.)'.19X.5F7.2)                    OPTYLD
    9 FORMAT(10F8.2)                                                      OPTYLD
   10 FORMAT(1H0.10X'MINIMUM CUTTING LENGTH (IN.)'.11X.F7.2)              OPTYLD
   11 FORMAT(1H0.10X.'MAXIMUM CUTTING LENGTH (IN.)'.11X.F7.2)             OPTYLD
   12 FORMAT(1H0.10X'CUTTING LENGTHS (IN.)'.4X.5F7.2/.36X.5F7.2)          OPTYLD
   13 FORMAT(1H0.10X.'SALVAGE LENGTHS'.24X.'RANDOM')                      OPTYLD
   14 FORMAT(1H0.10X.'VALUE MAXIMIZATION')                                OPTYLD
   15 FORMAT(1H0.10X.'YIELD MAXIMIZATION')                                OPTYLD
   16 FORMAT(9I0)                                                         OPTYLD
   17 FORMAT(5I0)                                                         OPTYLD
   18 FORMAT(///50X.'VALUE TABLE'///25X.9I9/15X.90(1H*)//)                OPTYLD
   20 FORMAT(24X.F4.2.'*'.7X.9I9/)                                        OPTYLD
   21 FORMAT(1H1.5X)                                                      OPTYLD
   22 FORMAT(I5.3A3.40X.I3.3X.I2.I4.I3.I4)                                OPTYLD
   23 FORMAT(1H0.3A3.'  BOARD NUMBER'.I7)                                 OPTYLD
   24 FORMAT(I5.14X.I2.40X.I6)                                            OPTYLD
   25 FORMAT(1H0.3X.'LENGTH ='.F7.2.'  WIDTH ='.F7.2)                     OPTYLD
   26 FORMAT('BOARD*'I5.'HAS MORE THAN 100 DEFECTS')                      OPTYLD
   27 FORMAT('*'.I5.'IS BOARD TOO WIDE')                                  OPTYLD
   28 FORMAT('*'.I5.'IS BOARD TOO LONG')                                  OPTYLD
   29 FORMAT(70X.I2)                                                      OPTYLD
   30 FORMAT(8X.3(A3.I2.I3.I4.I3.I3.I4))                                  OPTYLD
   31 FORMAT(15X.'USING GIVEN WIDTHS AND CONSTRAINTS. BOARD'.I7.          OPTYLD
     1       ' DOES NOT HAVE AN OPTIMUM SOLUTION')                        OPTYLD
   32 FORMAT(1H0.'PRIMARY CUTTINGS'.19X.'VALUE')                          OPTYLD
   33 FORMAT(1H0.'PRIMARY CUTTINGS'.15X.'AREA(SQ.FT.)')                   OPTYLD
   34 FORMAT(10X.F9.2.'   X'.F6.2.5X.F6.2)                                OPTYLD
   35 FORMAT(5X.'SURFACE AREA ='.F7.2.3X.'TOTAL VALUE='.F6.2)             OPTYLD
   36 FORMAT(5X.'SURFACE AREA ='.F7.2.3X.'TOTAL SQ.FT.='.F6.2)            OPTYLD
   37 FORMAT(10X.'X YIELD ='.F7.2)                                        OPTYLD
   38 FORMAT(1H0.'RERIP CUTTINGS  '.19X.'VALUE')                          OPTYLD
   39 FORMAT(1H0.'RERIP CUTTINGS  '.15X.'AREA(SQ.FT.)')                   OPTYLD
   40 FORMAT(1H0.15X.'BOARD AREA  ='.F10.2.'  SQ.IN.')                    OPTYLD
   41 FORMAT(16X.'TOTAL YIELD ='.F10.2.'  PERCENT')                       OPTYLD
   42 FORMAT(16X.'VALUE  '.F10.2)                                         OPTYLD
   43 FORMAT(16X.'AREA(SQ.FT.)'.F10.2)                                    OPTYLD
   44 FORMAT(//5X.'SAWCUT COORDINATES LX UX LY UY')                       OPTYLD
   45 FORMAT(2I5)                                                         OPTYLD
   46 FORMAT(4I10)                                                        OPTYLD
   47 FORMAT(//4X.'SALVAGE COORDINATES LX UX LY UY')                      OPTYLD
   48 FORMAT(15X.'INPUT OPTION ERROR')                                    OPTYLD
      END                                                                 OPTYLD
```

*** STATEMENT NUMBERS ***

```
   1     39  *270
   2     44    48    49    50  *271
   3     51  *272
   4     52    53  *273
   5     62    72  *274
   6     63  *275
   7     68  *276
   8     73  *277
   9     78  *278
  10     83  *279
  11     84  *280
  12     85  *281
  13     86  *282
  14     90  *283
  15     91  *284
  16     93    97  *285
  17     94  *286
  18     95  *287
  20     99  *288
  21    101  *289
  22    108  *290
  23    118  *291
  24    119  *292
  25    124  *293
  26    128  *294
  27    131  *295
  28    134  *296
  29    135  *297
  30    143  *298
  31    205  *299
  32    212  *301
  33    213  *302
  34    218   237  *303
  35    223   242  *304
  36    224  *305
  37    226   244  *306
  38    231  *307
  39    232  *308
  40    245  *309
  41    247  *310
  42    248  *311
  43    249  *312
  44    255  *313
  45    257  *314
  46    259   260   265  *315
  47    263  *316
  48    268  *317
 100    *30   100
 105     38   *43
 110     69   *71
 115     74   *76
 120     87   *89
 125     96  *100
 135     92  *106   138   206   253   262   267
 140    114  *116
 145    113  *117
 160    127  *130
 165    130  *133
 170    129   132  *135   136
 175    133  *139
 180    147   148  *150
 185    153  *159
```

```
190        159  *164
200        151   152   154   158   162  *168
205        140   141  *169
210        139  *172
220        190  *193
225        189  *194
230        185  *200
235        173  *201
240        172  *204
250        214  *221
260        211  *227
265        233  *240
270        230  *245
280        258  *261
285        254  *262
290        264  *266
295         60    61  *268
300        112  *269

                        ***  VARIABLES  ***

A            6  *207  *219   223   224   225  *227  *230   242   243
AREA         6    12  *121   225   243   245   250
AVAIL       17   254   256
BDLX        24  *108   120
BDLY        24  *108   121   123
BDNUM       16  *108   112   118  *119   128   131   134   205   250   251   257
BDUX        24  *108   120   133
BDUY        24  *108   121   123   130
BIT       *187   192
BOARD       16  *115   192
CLASS       10  *143   148
COLECT     204
DLX         10  *143   176   182
DLY1        10  *143   161   166   174   181
DLY2        10  *143   157   164   165  *166   181
DOLMAX       9    12   *55   *59    90    91    92   212   213   223   224   231
           232   240   249
DUX         10  *143   177   182
DUY1        10  *143   153  *161  *164   175   180
DUY2        10  *143   154  *157   159  *165   180
FIRST      *37   106  *109
FLD       *192
FLOAT       47   178   179   216   217   235   236
FOR3        10   *31   *64   *65    66    67
FOR4        10   *32    64    65
GR1          6  *108   118
GR2          6  *108   118
GR3          6  *108   118
GRID         6    15   *45    47    70    75    88    98   121   122   123   216
           217   235   236
I          *38    41    42   *62    68   *69    70   *72    73   *74    75   *78
            85   *87    88   *93   *94    95   *97    99  *113   115  *143  *147
           148  *151   152   153   154   157   159   161   164   165   166  *185
           186   187  *214   215   216   217  *233   234   235   236   251  *258
           259   260  *264   265
I1        *125   141  *142  *143   146  *149   151
I2        *140   141   142  *143   145   151
IABS       187
IFIX        70    75    81    82   196   197
INT         88
J          *96   *97    98    99  *114   115  *190   191   251   259   260   265
J1        *146   147
J2        *145   147
K         *173   174   175   176   177   180   181   182
K1        *191   192
K2        *186   187  *188   192
KERF         6    13   *46

L            6  *122   124
LENGTH      20   *80
LSLOPE       6  *184   194
LTEMP        6  *216   218   219  *235   237   238
LVALUE      23   *33   *93    95
MAX         12   246   247   250
MAXLEN      16   *82
MINLEN      16   *81
NARROW      15   *44    47
ND        *108   127   139   140   152   172   173
NLEN        20   *51    60   *78    80    85    87
NP          12  *120   121   122   205   206
NPIECE      18   211   214   250   251
NREWID      21   *50    61   *72    73    74
NSAW        13   *40    60
NSEC        14  *111   230   233   250   251   262   264
NSEQ      *135   136
NUNIT     *119
NWIDTH      13   *49    60   *62    68    69
OPTION      10   *41   *42    57    58    59
PIECE       18   215   216   217   251
PRTOUT       9   *54   *58   118   124   212   213   218   223   224   226   231
           232   237   242   244   245   247   248   249   253
RANDOM       9    22   *56   *57    65    66    67    79    80    83    84    85
RERIP       14   234   235   236   251
REWID       21   *75    77
RISE1        6  *180   183
RISE2        6  *181   184
RTEMP        6   *47    63
RUN       *182   183   184
SAL         14   265
SALMIN      12   *77
SALVAG      21
SAWCUT      17   259   260
SAWMAX       6   *53   *80    82    84
SAWMIN       6   *52   *79    81    83
SM        *119   250
T         *256   257   258
TEMP       *39    40    41    42
TEMWID       6   *62    68    70   *72    73    75   *78    79    80    85    88
TEXT        10   *30    66    67
UNIT       *25   108  *110   119  *137   143
UNIT1      *26   250   251
UNIT2      *27    39    44    48    49    50    51    52    53    62    72    78
            93    94    97
UNIT3      *28   257   260
URTIMG       6   106
URTIMS       6   107
USLOPE       6  *183   195
V            6  *208  *220   222   223   224  *228  *239   241   242
VALUE        6   215   234
VALUES      23   *29   *97    99
VTEMP        6  *215   218   220  *234   237   239
W            6  *123   124
WIDTH       19   *70
WORTH        6  *210  *222  *241   248   249   250
WTEMP        6   *98    99  *217   218   219  *236   237   238
WVALUE      23   *34   *94    98
X1        *176   185
X2        *177   185
Y1        *174   178   190  *196  *198
Y2        *175   179   189   198  *197  *199
YIELD        6    12  *246   247   250
YLD          6  *209  *225   226  *229  *243   244   246
YTEMP1       6  *178  *194   196
YTEMP2       6  *179  *195   197
```

```
 1        SUBROUTINE COLECT                                              COLECT
 2        IMPLICIT INTEGER(A-Z)                                          COLECT
 3        REAL GRID,YIELD,AREA,CLRYLD,WORTH,BDWID,VALUE,VTEMP            COLECT
 4        LOGICAL DOLMAX,FOUND                                           COLECT
 5        DIMENSION WORTH(2)                                             COLECT
 6        COMMON /MD/ MAX,YIELD(2),NP,AREA,DOLMAX,SALMIN                 COLECT
 7        COMMON /MDSP/ GRID,NARROW                                      COLECT
 8        COMMON /MNOV/ SAWCUT(100,4),AVAIL                              COLECT
 9        COMMON /MDCV/ NPIECE,PIECE(50,2)                               COLECT
10        COMMON /DSCP/ COMB,YLOW,YHI,RIPCOM(2,8)                        COLECT
11        COMMON/OSC /ACTIVE,XLOW,XHI                                    COLECT
12        COMMON /OPS/ IRIP(8),NR(2)                                     COLECT
13        COMMON /OSCHP/FOUND                                            COLECT
14        COMMON /MOD/BDLX,BDUX,BDLY,BDUY                                COLECT
15 C *** ASSIGN INITIAL CONDITIONS                                       COLECT
16        BDWID=((BDUY-BDLY)-NARROW)*GRID                                COLECT
17        ACTIVE=1                                                       COLECT
18        MAX=2                                                          COLECT
19        DO 210 J=1,8                                                   COLECT
20          RIPCOM(1,J)=1                                                COLECT
21    210 CONTINUE                                                       COLECT
22        YIELD(MAX)=.0001                                               COLECT
23        WORTH(MAX)=.0001                                               COLECT
24        NR(MAX)=0                                                      COLECT
25        FOUND=.FALSE.                                                  COLECT
26        CALL INITAL($220,$255,BDWID,ACTIVE)                            COLECT
27    215 CALL PERMU($220,$255,BDWID,ACTIVE)                             COLECT
28    220 AVAIL=1                                                        COLECT
29        NPIECE=0                                                       COLECT
30        YLOW=BDLY+1                                                    COLECT
31        YLOW=YLOW+NARROW                                               COLECT
32        XLOW=BDLX+1                                                    COLECT
33        XHI=BDUX                                                       COLECT
34        CALL SAW(BDUY)                                                 COLECT
35        TOP=BDUY                                                       COLECT
36        IF(.NOT.FOUND)GO TO 230                                        COLECT
37        IF(YHI.EQ.TOP) GO TO 230                                       COLECT
38        YLOW=YHI+2                                                     COLECT
39        YHI=TOP                                                        COLECT
40        IF((YHI-YLOW).LT.SALMIN) GO TO 230                             COLECT
41        CALL NOCHIP (BDLX+1,BDUX,YLOW,YHI)                             COLECT
42    230 CLRYLD=0.                                                      COLECT
43        WORTH(ACTIVE)=0.0                                              COLECT
44        IF(NPIECE.EQ.0)GO TO 240                                       COLECT
45        DO 235 I=1,NPIECE                                              COLECT
46          CLRYLD=CLRYLD+PIECE(I,1)*PIECE(I,2)                          COLECT
47          VTEMP=VALUE(PIECE(I,1),PIECE(I,2))                           COLECT
48          WORTH(ACTIVE)=WORTH(ACTIVE)+VTEMP                            COLECT
49    235 CONTINUE                                                       COLECT
50    240 YIELD(ACTIVE)=(CLRYLD*(GRID**2)/AREA)*100.                     COLECT
51        IF(FOUND.AND.COMB.LE.1)GO TO 245                               COLECT
52        IF(FOUND)GO TO 255                                             COLECT
53 C *** TEST IF ACTIVE COMBINATION EXCEEDS PRESENT MAXIMUM              COLECT
54        IF(.NOT.DOLMAX.AND.YIELD(ACTIVE).LT.YIELD(MAX))GO TO 215       COLECT
55        IF(DOLMAX.AND.WORTH(ACTIVE).LT.WORTH(MAX))GO TO 215            COLECT
56    245 TEMP=ACTIVE                                                    COLECT
57        NRTEMP=NR(ACTIVE)                                              COLECT
58        ACTIVE=MAX                                                     COLECT
59        MAX=TEMP                                                       COLECT
60        DO 250 L=1,NRTEMP                                              COLECT
61          RIPCOM(ACTIVE,L)=RIPCOM(MAX,L)                               COLECT
62    250 CONTINUE                                                       COLECT
63        IF(FOUND)GO TO 255                                             COLECT
64        GO TO 215                                                      COLECT
65    255 IF(COMB.LE.-1)NP=-1                                            COLECT
66        IF(FOUND)RETURN                                                COLECT
67        FOUND=.TRUE.                                                   COLECT
68        ACTIVE=MAX                                                     COLECT
69        GO TO 220                                                      COLECT
70        END                                                            COLECT
```

```
                    *** STATEMENT NUMBERS ***

 210      19   *21
 215     *27    54    55    64
 220      26    27   *28    69
 230      36    37    40   *42
 235      45   *49
 240      44   *50
 245      51   *56
 250      60   *62
 255      26    27    52    63   *65

                    *** VARIABLES ***

ACTIVE    11   *17    26    27    43    48    50    54    55    56    57   *58
          61   *68
AREA       3     6    50
AVAIL      8   *28
BDLX      14    32    41
BDLY      14    16    30
BDUX      14    33    41
BDUY      14    16    34    35
BDWID      3   *16    26    27
CLRYLD     3   *42   *46    50
COLECT     1
COMB      10    51    65
DOLMAX     4     6    54    55
FOUND      4    13   *25    36    51    52    63    66   *67
GRID       3     7    16    50
I        *45    46    47
INITAL    26
IRIP      12
J        *19    20
L        *60    61
MAX        6   *18    22    23    24    54    55    58   *59    61    68
NARROW     7    16    31
NOCHIP    41
NP         6   *65
NPIECE     9   *29    44    45
NR        12   *24    57
NRTEMP   *57    60
PERMU     27
PIECE      9    46    47
RIPCOM    10   *20   *61
SALMIN     6    40
SAW       34
SAWCUT     8
TEMP     *56    59
TOP      *35    37    39
VALUE      3    47
VTEMP      3   *47    48
WORTH      3     5   *23   *43   *48    55
XHI       11   *33
XLOW      11   *32
YHI       10    37    38   *39    40    41
YIELD      3     6   *22   *50    54
YLOW      10   *30   *31   *38    40    41
```

```
 1        SUBROUTINE PERM2                                               PERMU
 2 C *** CALCULATES NEXT RIP COMBINATION                                 PERMU
 3        IMPLICIT INTEGER(A-Z)                                          PERMU
 4        REAL WIDTH(5),KERF,SUM,MINUSE,BDWID,GRID                       PERMU
 5        LOGICAL OK,FOUND                                               PERMU
 6        COMMON /MDSP/GRID,NARROW                                       PERMU
 7        COMMON /MP/ NWIDTH,NSAW,KERF                                   PERMU
```

```
 8       COMMON /MPSC/TWIDTH(5)                                    PERMU
 9       COMMON /OPS/ IRIP(8),NR(2)                                PERMU
10       COMMON /OSCP/COMB,YLOW,YHI,RIPCOM(2,8)                    PERMU
11       COMMON /OSCHP/FOUND                                       PERMU
12       ENTRY INITAL(*,$,BDWID,ACTIVE)                            PERMU
13       DO 305 I=1,NWIDTH                                         PERMU
14        WIDTH(I)=FLOAT(TWIDTH(I))*GRID                           PERMU
15   305 CONTINUE                                                  PERMU
16 C *** CALCULATE THE NUMBER OF SAWS NEEDED                        PERMU
17       MSAW=0                                                    PERMU
18         LIMIT=NSAW                                              PERMU
19   312   SUM=0.                                                  PERMU
20         DO 315 I=1,LIMIT                                        PERMU
21           SUM=SUM+WIDTH(I)                                      PERMU
22   315   CONTINUE                                                PERMU
23         IF(SUM.GT.BDWID)LIMIT=LIMIT-1                           PERMU
24         IF(LIMIT.EQ.0)GO TO 320                                 PERMU
25         IF(LIMIT.EQ.0)GO TO 320                                 PERMU
26         IF(SUM.GT.BDWID)GO TO 312                               PERMU
27         IF(LIMIT.GT.MSAW)MSAW=LIMIT                             PERMU
28   320 IF(MSAW.EQ.0)RETURN 2                                     PERMU
29       OLDSAW=MSAW                                               PERMU
30       QUIT=0                                                    PERMU
31       DO 325 K=1,MSAW                                           PERMU
32         IRIP(K)=1                                               PERMU
33   325 CONTINUE                                                  PERMU
34       COMB=0                                                    PERMU
35       LAST=MSAW                                                 PERMU
36       NR(ACTIVE)=0                                              PERMU
37       GO TO 350                                                 PERMU
38       ENTRY PERMU(*,$,BDWID,ACTIVE)                             PERMU
39   330 OK=.FALSE.                                                PERMU
40 C *** GENERATE RIP COMBINATION                                   PERMU
41       LAST=MSAW                                                 PERMU
42       DO 335 J=LAST,1,-1                                        PERMU
43         IF(OK)GO TO 335                                         PERMU
44         IRIP(J)=IRIP(J)+1                                       PERMU
45         IF(IRIP(J).LE.NWIDTH)OK=.TRUE.                          PERMU
46         IF(IRIP(J).LE.NWIDTH)GO TO 335                          PERMU
47         IF(J.EQ.1)QUIT=1                                        PERMU
48         IRIP(J)=1                                               PERMU
49   335 CONTINUE                                                  PERMU
50       IF(QUIT.LE.0)GO TO 350                                    PERMU
51       IF(COMB.GT.0)RETURN 2                                     PERMU
52       COMB=-1                                                   PERMU
53       FOUND=.TRUE.                                              PERMU
54       RETURN 2                                                  PERMU
55 C *** TEST VALIDITY OF RIP COMBINATION                            PERMU
56   350 SUM=0.                                                    PERMU
57       DO 355 K=1,LAST                                           PERMU
58         TEMP=IRIP(K)                                            PERMU
59         IF(K.NE.LAST)SUM=SUM+WIDTH(TEMP)+KERF                   PERMU
60         IF(K.EQ.LAST)SUM=SUM+WIDTH(TEMP)                        PERMU
61   355 CONTINUE                                                  PERMU
62   360 TEMP=IRIP(1)                                              PERMU
63       IF(LAST.EQ.1)SUM=WIDTH(TEMP)                              PERMU
64       IF(SUM.LE.BDWID)GO TO 365                                 PERMU
65       LAST=LAST-1                                               PERMU
66       IF(LAST.LE.0)RETURN 2                                     PERMU
67       GO TO 350                                                 PERMU
68   365 MINUSE=(BDWID+KERF)-SUM                                   PERMU
69       IF(MINUSE.LT.0)GO TO 360                                  PERMU
70       IF(QUIT.LT.1 .AND.MINUSE.GT.WIDTH(1)+KERF)GO TO 330       PERMU
71       IF(QUIT.GE.1)FOUND=.TRUE.                                 PERMU
72       IF(NR(ACTIVE).EQ.LAST)GO TO 370                           PERMU
73       COMB=COMB+1                                               PERMU
74       IF(COMB.EQ.1)MSAW=LAST                                    PERMU
75       NR(ACTIVE)=LAST                                           PERMU
76       RETURN 1                                                  PERMU
77   370 OUT=0                                                     PERMU
78       DO 375 I=1,LAST                                           PERMU
79       IF(IRIP(I).EQ.RIPCOM(ACTIVE,I))OUT=OUT+1                  PERMU
80   375 CONTINUE                                                  PERMU
81       IF(OUT.GE.LAST)GO TO 330                                  PERMU
82       COMB=COMB+1                                               PERMU
83       RETURN 1                                                  PERMU
84       END                                                       PERMU
```

*** STATEMENT NUMBERS ***

```
305     13   *15
312    *19    26
315     20   *22
320     24    25   *28
325     31   *33
330    *39    70    81
335     42    43    46   *49
350     37    50   *56    67
355     57   *61
360    *62    69
365     64   *68
370     72   *77
375     78   *80
```

*** VARIABLES ***

```
ACTIVE    12    36    38    72    75    79
BDWID      4    12    23    26    38    64    68
COMB      10   *34    51   *52   *73    74   *82
FLOAT     14
FOUND      5    11   *53   *71
GRID       4     6    14
I        *13    14   *20   *78    79
INITAL    12
IRIP       9   *32   *44    45    46   *48    58    62    79
J        *42    44    45    46    47    48
K        *31    32   *57    58    59    60
KERF       4     7    59    68    70
LAST     *35   *41    42    57    59    60    63   *65    66    72    74    75
          78    81
LIMIT    *18    20   *23    24    25    27
MINUSE     4   *68    69    70
MSAW     *17   *27    28    29    31    35    41   *74
NARROW     6
NR         9   *36    72   *75
NSAW       7    18
NWIDTH     7    13    45    46
OK         5   *39    43   *45
OLDSAW   *29
OUT      *77   *79    81
PERM2      1
PERMU     38
QUIT     *30   *47    50    70    71
RIPCOM    10    79
SUM        4   *19   *21    23    26   *56   *59   *60   *63    64    68
TEMP     *58    59    60   *62    63
TWIDTH     8    14
WIDTH      4   *14    21    59    60    63    70
YHI       10
YLOW      10
```

```
 1        SUBROUTINE SAW(BDUY)                                            SAW
 2        IMPLICIT INTEGER (A-Z)                                          SAW
 3        REAL GRID                                                       SAW
 4        DIMENSION TRIP(8)                                               SAW
 5        LOGICAL FJ.RANDOM.FOUND.REJECT                                  SAW
 6        COMMON /MOSP/GRID.NARROW                                        SAW
 7        COMMON /OSCP/ COMB.YLOW.YHI.RIPCOM(2.8)                         SAW
 8        COMMON /MPSC/ WIDTH(5)                                          SAW
 9        COMMON /MSD/ BOARD(100.24).MAXLEN.BDNUM.MINLEN                  SAW
10        COMMON /OPS/ IRIP(8).NRTEMP(2)                                  SAW
11        COMMON /OSC /ACTIVE.XLOW.XHI                                    SAW
12        COMMON /SC/ I.SAVSAW.BEG.END                                    SAW
13        COMMON /MCS/RANDOM                                              SAW
14        COMMON /SD/ FJ                                                  SAW
15        COMMON /SV/RIPWID(8)                                            SAW
16        COMMON /SCV/ RIP                                                SAW
17        COMMON /OSCHP/FOUND                                             SAW
18 C *** ASSIGN RIP COMBINATIONS AND CHECK FOR PREVIOUSLY GENERATED RIPS   SAW
19        REJECT=.FALSE.                                                  SAW
20        FJ=.FALSE.                                                      SAW
21        NR=NRTEMP(ACTIVE)                                               SAW
22        IF(RANDOM.AND.MAXLEN.GT.1)FJ=.TRUE.                             SAW
23        START=1                                                         SAW
24        RIP=1                                                           SAW
25        IF(FOUND)REJECT=.TRUE.                                          SAW
26        IF(FOUND.AND. COMB.GT.1)GO TO 410                               SAW
27        DO 405 J=1.NR                                                   SAW
28          TEMP=IRIP(J)                                                  SAW
29          TRIP(J)=RIPCOM(ACTIVE.J)                                      SAW
30          RIPCOM(ACTIVE.J)=IRIP(J)                                      SAW
31    405 CONTINUE                                                        SAW
32    410 DO 420 J=1.NR                                                   SAW
33          IF(REJECT OR.COMB.LE.1)GO TO 415                              SAW
34          IF(RIPCOM(ACTIVE.J).EQ.TRIP(J))GO TO 415                      SAW
35          REJECT=.TRUE.                                                 SAW
36          START=J                                                       SAW
37    415   TEMP=RIPCOM(ACTIVE.J)                                         SAW
38          RIPWID(J)=WIDTH(TEMP)                                         SAW
39    420 CONTINUE                                                        SAW
40        IF(NARROW.LE.0)GO TO 425                                        SAW
41        EDGE=YLOW-NARROW                                                SAW
42        CALL STORE(XLOW-1.XHI.EDGE-NARROW.EDGE)                         SAW
43    425 IF(START.EQ.1)GO TO 435                                         SAW
44        IF(.NOT.FOUND)CALL RETREV(START)                                SAW
45        LAST=START-1                                                    SAW
46        DO 430 I=1.LAST                                                 SAW
47          TEMP=RIPCOM(ACTIVE.I)                                         SAW
48          YHI=YLOW+WIDTH(TEMP)-1                                        SAW
49          YLOW=YHI+2                                                    SAW
50    430 CONTINUE                                                        SAW
51 C *** RIP THE BOARD                                                     SAW
52    435 DO 465 RIP=START.NR                                             SAW
53          TEMP=RIPCOM(ACTIVE.RIP)                                       SAW
54          YHI=YLOW+WIDTH(TEMP)-1                                        SAW
55          CALL STORE(XLOW-1.XHI.YHI.YHI+1)                              SAW
56          BEG=XLOW                                                      SAW
57          END=XLOW-1                                                    SAW
58          CLRKNT=0                                                      SAW
59          SAVSAW=XLOW-1                                                 SAW
60          K2=1                                                          SAW
61 C *** SCAN THE RIP FOR DEFECTS AND CLEAR AREAS                          SAW
62          DO 460 I=XLOW.XHI                                             SAW
63            K2=I/36                                                     SAW
64            BIT=IABS(I-K2*36)                                           SAW
65            K2=K2+1                                                     SAW
66            DO 445 J=YLOW.YHI                                           SAW
67              K1=J                                                      SAW
68              IF(FLD(BIT.1.BOARD(K1.K2)).EQ.0)GO TO 445                 SAW
69              GO TO 450                                                 SAW
70    445         CONTINUE                                                SAW
71                CLRKNT=CLRKNT+                                          SAW
72                IF(FJ.AND.CLRKNT.GE.MAXLEN)GO TO 450                    SAW
73                GO TO 460                                               SAW
74 C                                                                      SAW
75 C *** DEFECT FOUND                                                      SAW
76 C                                                                      SAW
77    450         IF(CLRKNT.LT.MINLEN) GO TO 455                          SAW
78                CALL CROSS                                              SAW
79                BEG=I                                                   SAW
80    455         END=I                                                   SAW
81                CLRKNT=0                                                SAW
82    460       CONTINUE                                                  SAW
83              I=XHI+1                                                   SAW
84              IF(CLRKNT.LT.MINLEN) END=XHI                              SAW
85              CALL CROSS                                                SAW
86              YLOW=YHI+2                                                SAW
87    465   CONTINUE                                                      SAW
88          RETURN                                                        SAW
89          END                                                           SAW
```

*** STATEMENT NUMBERS ***

405	27	*31		
410	26	*32		
415	33	34	*37	
420	32	*39		
425	40	*43		
430	46	*50		
435	43	*52		
445	66	68	*70	
450	69	72	*77	
455	77	*80		
460	62	73	*82	
465	52	*87		

*** VARIABLES ***

ACTIVE	11	21	29	30	34	37	47	53			
BDNUM	9										
BDUY	1										
BEG	12	*56	*79								
BIT	*64	68									
BOARD	9	68									
CLRKNT	*58	*71	72	77	*81	84					
COMB	7	26	33								
CROSS	78	85									
EDGE	*41	42									
END	12	*57	*80	*84							
FJ	5	14	*20	*22	72						
FLD	68										
FOUND	5	17	25	26	44						
GRID	3	6									
I	12	*46	47	*62	63	64	79	80	*83		
IABS	64										
IRIP	10	28	30								
J	*27	28	29	30	*32	34	36	37	38	*66	67
K1	*67	68									
K2	*60	*63	64	*65	68						
LAST	*45	46									
MAXLEN	9	22	72								
MINLEN	9	77	84								
NARROW	6	40	41	42							
NR	*21	27	32	52							
NRTEMP	10	21									
RANDOM	5	13	22								
REJECT	5	*19	*25	33	*35						
RETREV	44										
RIP	16	*24	*52	53							

```
RIPCOM       7    29   *30    34    37    47    53
RIPWID      15   *38
SAVSAW      12   *59
SAW          1
START      *23   *36    43    44    45    52
STORE       42    55
TEMP       *28   *37    38   *47    48   *53    54
TPIP         4   *29    34
WIDTH        8    38    48    54
XHI         11    42    55    62    83    84
XLOW        11    42    55    56    57    59    62
YHI          7   *48    49   *54    55    66    86
YLOW         7    41    48   *49    54    66   *86

 1         SUBROUTINE CROSS                                          CROSS
 2         IMPLICIT INTEGER(A-Z)                                     CROSS
 3         REAL CLRLEN                                               CROSS
 4         LOGICAL RANDOM,FOUND                                      CROSS
 5         COMMON /MC/LENGTH(10),NLEN                                CROSS
 6         COMMON /MDCV/ NPIECE,PIECE(50,2)                          CROSS
 7         COMMON /OSCP/ COMB,YLOW,YHI,RIPCOM(2,8)                   CROSS
 8         COMMON /OSC/ ACTIVE,XLOW,XHI                              CROSS
 9         COMMON /MPSC/ WIDTH(5)                                    CROSS
10         COMMON /SC/ I,SAVSAW,BEG,END                              CROSS
11         COMMON /MCS/ RANDOM                                       CROSS
12         COMMON /SCV/ RIP                                          CROSS
13         COMMON /OSCHP/FOUND                                       CROSS
14 C *** CHECK IF A PREVIOUS CUTTING OR DEFECT OCCURRED               CROSS
15         IF(END.EQ.(LOW-1)) GO TO 110                              CROSS
16 C *** CHECK IF PREVIOUS CUTTINGS SHOULD BE RERIPPED                CROSS
17         IF(END.NE.BEG)GO TO510                                    CROSS
18         IF((BEG.EQ.XHI).AND.(.NOT.RANDOM)) GO TO 530              CROSS
19         GO TO 525                                                 CROSS
20   510   IF(.NOT.RANDOM) GO TO 520                                 CROSS
21         IF(.NOT.FOUND)GO TO515                                    CROSS
22         TEMPL=BEG                                                 CROSS
23         TEMPH=END-1                                               CROSS
24         IF(BEG.EQ.XLOW)TEMPL=XLOW-1                               CROSS
25         IF(END.EQ.XHI)TEMPH=TEMPH+1                               CROSS
26         CALL NOCHIP (TEMPL,TEMPH,YLOW-1,YHI)                      CROSS
27 C *** PLACE CROSSCUT AT BEGINNING OF PREVIOUS DEFECT                CROSS
28   515   IF(BEG.NE.XLOW)CALL STORE(BEG-1,BEG,YLOW-1,YHI)           CROSS
29   520   IF(END.EQ.XHI) GO TO 550                                  CROSS
30   525   CALL STORE(END-1,END,YLOW-1,YHI)                          CROSS
31   530   IF(END.EQ.XHI) GO TO 550                                  CROSS
32 C                                                                 CROSS
33 C *** RANDOM LENGTH CUTTINGS                                       CROSS
34 C                                                                 CROSS
35   110   TEMP=RIPCOM(ACTIVE,RIP)                                   CROSS
36         IF(.NOT.RANDOM) GO TO 535                                 CROSS
37         CLRLEN=((I-END)-1)                                        CROSS
38         NPIECE=NPIECE+1                                           CROSS
39         IF(NPIECE.GT.50) GO TO 570                                CROSS
40         PIECE(NPIECE,1)=CLRLEN                                    CROSS
41         PIECE(NPIECE,2)=WIDTH(TEMP)                               CROSS
42         IF(.NOT.FOUND)CALL VALSTO                                 CROSS
43         RETURN                                                    CROSS
44 C                                                                 CROSS
45 C *** SPECIFIED LENGTH CUTTINGS                                    CROSS
46 C                                                                 CROSS
47   535   CLRGRD=(I-END)-1                                          CROSS
48         START=(END+1)                                             CROSS
49   540   DO 545 J=NLEN,1,-1                                        CROSS
50         IF(CLRGRD.GE.(LENGTH(J))) GO TO 560                       CROSS
51   545   CONTINUE                                                  CROSS
52   550   IF(RANDOM)RETURN                                          CROSS
53         IF(FOUND)CALL NOCHIP (SAVSAW,END,YLOW-1,YHI)              CROSS
54         SAVSAW=TEMPSW                                             CROSS
55         RETURN                                                    CROSS
56 C *** REMOVE LONGEST SPECIFIED LENGTH AND TRY TO CUT REMAINDER     CROSS
57   560   CLRGRD=(CLRGRD-(LENGTH(J)))-1                             CROSS
58         CP=START+(LENGTH(J))                                      CROSS
59         IF(CP.EQ.XHI+1) GO TO 565                                 CROSS
60         CALL STORE(CP-1,CP,YLOW-1,YHI)                            CROSS
61         TEMPSW=CP                                                 CROSS
62         START=CP+1                                                CROSS
63   565   NPIECE=NPIECE+1                                           CROSS
64         IF(NPIECE.GT.50) GO TO 570                                CROSS
65         PIECE(NPIECE,1)=LENGTH(J)                                 CROSS
66         PIECE(NPIECE,2)=WIDTH(TEMP)                               CROSS
67         IF(.NOT.FOUND)CALL VALSTO                                 CROSS
68         GO TO 540                                                 CROSS
69 C *** NUMBER OF CUTTINGS EXCEED ARRAY SIZE                         CROSS
70   570   NPIECE=50                                                 CROSS
71         WRITE(6,575)                                              CROSS
72   575   FORMAT(' MORE THAN 50 PRIMARY CUTTINGS')                  CROSS
73         RETURN                                                    CROSS
74         END                                                       CROSS

                    *** STATEMENT NUMBERS ***

   110      15   *35
   510      17   *20
   515      21   *28
   520      20   *29
   525      19   *30
   530      18   *31
   535      36   *47
   540     *49    68
   545      49   *51
   550      29    31   *52
   560      50   *57
   565      59   *63
   570      39    64   *70
   575      71   *72

                    *** VARIABLES ***

ACTIVE       8
BEG         10    17    18    22    24    28
CLRGRD     *47    50   *57
CLRLEN       3   *37    40
COMB         7
CP         *58    59    60    61    62
CROSS        1
END         10    15    17    23    25    29    30    31    37    47    48    53
FOUND        4    13    21    42    53    67
I           10    37    47
J          *49    50    57    58    65
LENGTH       5    50    57    58    65
NLEN         5    49
NOCHIP      26    53
NPIECE       6   *38    39    40    41   *63    64    65    66   *70
PIECE        6   *40   *41   *65   *66
RANDOM       4    11    18    20    36    52
RIP         12    35
RIPCOM       7    35
SAVSAW      10    53   *54
START      *48    58   *62
STORE       28    30    60
TEMP       *35    41    66
TEMPH      *23   *25    26
TEMPL      *22   *24    26
TEMPSW      54   *61
VALSTO      42    67
WIDTH        9    41    66
XHI          8    18    25    29    31    59
```

```
:LOW         8    15    24    28
:HI          7    26    28    30    53    60
:LOW         7    26    28    30    53    60
  1          SUBROUTINE VALSTO                                              VALSTO
  2 C ***    STORES PRIMARY CUTTING & RIP INFORMATION                       VALSTO
  3          IMPLICIT INTEGER (A-Z)                                         VALSTO
  4          DIMENSION PRIM(50.8).NINDEX(8).SAWC(30.10.4).SINDEX(10)        VALSTO
  5          COMMON /MOCV/ NPIECE.PIECE(50.2)                               VALSTO
  6          COMMON /SV/ RIPWID(8)                                          VALSTO
  7          COMMON /SCV/ RIP                                               VALSTO
  8          COMMON /MHOV/SAWCUT(100.4).AVAIL                               VALSTO
  9 C ***    **********STORE PRIMARY CUTTINGS INFORMATION *********** ***   VALSTO
 10          IF(NPIECE.NE.1)GO TO 620                                       VALSTO
 11          INDEX=0                                                        VALSTO
 12          OLDRIP=1                                                       VALSTO
 13          DO 615 J=1.50                                                  VALSTO
 14            DO 610 K=1.8                                                 VALSTO
 15              PRIM(J.K)=0.                                               VALSTO
 16              IF(J.EQ.1)NINDEX(K)=0                                      VALSTO
 17    610     CONTINUE                                                     VALSTO
 18    615   CONTINUE                                                       VALSTO
 19    620   IF(RIP.NE.OLDRIP)INDEX=0                                       VALSTO
 20          INDEX=INDEX+1                                                  VALSTO
 21          IF(INDEX.GT.50)WRITE(6.625)                                    VALSTO
 22   625    FORMAT(20X.'MORE THAN 50 CUTTINGS PER RIP')                    VALSTO
 23          IF(INDEX.GT.50)RETURN                                          VALSTO
 24          PRIM(INDEX.RIP)=PIECE(NPIECE.1)                                VALSTO
 25          NINDEX(RIP)=INDEX                                              VALSTO
 26          OLDRIP=RIP                                                     VALSTO
 27          RETURN                                                         VALSTO
 28 C ***    **********STORE SAWCUTS *********** ***                        VALSTO
 29          ENTRY SAWSTO (CUT1.CUT2.CUT3.CUT4)                             VALSTO
 30          IF(AVAIL.NE.1)GO TO 645                                        VALSTO
 31          SAWRIP=RIPWID(RIP)                                             VALSTO
 32          NSRIP=1                                                        VALSTO
 33          SAWIND=0                                                       VALSTO
 34          DO 640 J=1.25                                                  VALSTO
 35            DO 635 K=1.10                                                VALSTO
 36              DO 630 I=1.4                                               VALSTO
 37                SAWC(J.K.I)=0                                            VALSTO
 38    630       CONTINUE                                                   VALSTO
 39              IF(J.EQ.1)SINDEX(K)=0                                      VALSTO
 40    635     CONTINUE                                                     VALSTO
 41    640   CONTINUE                                                       VALSTO
 42    645   IF(RIPWID(RIP).NE.SAWRIP)SAWIND=0                              VALSTO
 43          IF(RIP.NE.NSRIP)SAWIND=0                                       VALSTO
 44          SAWIND=SAWIND+1                                                VALSTO
 45          IF(SAWIND.GT.30)WRITE(6.650)                                   VALSTO
 46    650   FORMAT(20X.'MORE THAN 30 CROSSCUTS PER RIP')                   VALSTO
 47          IF(SAWIND.GT.30)RETURN                                         VALSTO
 48          SAWC(SAWIND.RIP.1)=CUT1                                        VALSTO
 49          SAWC(SAWIND.RIP.2)=CUT2                                        VALSTO
 50          SAWC(SAWIND.RIP.3)=CUT3                                        VALSTO
 51          SAWC(SAWIND.RIP.4)=CUT4                                        VALSTO
 52          SINDEX(RIP)=SAWIND                                             VALSTO
 53          SAWRIP=RIPWID(RIP)                                             VALSTO
 54          NSRIP=RIP                                                      VALSTO
 55          RETURN                                                         VALSTO
 56 C ***    ********** RETREIVE STORED INFORMATION *********** ***          VALSTO
 57          ENTRY RETREV(START)                                            VALSTO
 58 C ***    RETREIVE PRIMARY CUTTINGS                                      VALSTO
 59          LAST=START-1                                                   VALSTO
 60          DO 680 J=1.LAST                                                VALSTO
 61            IF(NINDEX(J).EQ.0)GO TO 665                                  VALSTO
 62            NTEMP=NINDEX(J)                                              VALSTO
 63            DO 660 K=1.NTEMP                                             VALSTO
 64              NPIECE=NPIECE+1                                            VALSTO
 65              PIECE(NPIECE.1)=PRIM(K.J)                                  VALSTO
 66              PIECE(NPIECE.2)=RIPWID(J)                                  VALSTO
 67    660     CONTINUE                                                     VALSTO
 68    665   IF(SINDEX(J).EQ.0)GO TO 680                                    VALSTO
 69 C ***    RETREIVE SAWCUTS                                               VALSTO
 70          STEMP=SINDEX(J)                                                VALSTO
 71          DO 675 K=1.STEMP                                               VALSTO
 72            DO 670 I=1.4                                                 VALSTO
 73              SAWCUT(K.I)=SAWC(K.J.I)                                    VALSTO
 74    670     CONTINUE                                                     VALSTO
 75            AVAIL=AVAIL+1                                                VALSTO
 76    675   CONTINUE                                                       VALSTO
 77    680   CONTINUE                                                       VALSTO
 78          RETURN                                                         VALSTO
 79          END                                                            VALSTO
```

*** STATEMENT NUMBERS ***

```
   610      14    *17
   615      13    *18
   620      10    *19
   625      21    *22
   630      36    *38
   635      35    *40
   640      34    *41
   645      30    *42
   650      45    *46
   660      63    *67
   665      61    *68
   670      72    *74
   675      71    *76
   680      60     68    *77
```

*** VARIABLES ***

```
AVAIL      8    30   *75
CUT1      29    48
CUT2      29    49
CUT3      29    50
CUT4      29    51
I        *36    37   *72    73
INDEX    *11   *19   *20    21    23    24    25
J        *13    15    16   *34    37    39   *60    61    62    65    66    68
          70    73
K        *14    15    16   *35    37    39   *63    65   *71    73
LAST     *59    60
NINDEX     4   *16   *25    61    62
NPIECE     5    10    24   *64    65    66
NSRIP    *32    43   *54
NTEMP    *62    63
OLDRIP   *12    19   *26
PIECE      5    24   *65   *66
PRIM       4   *15   *24    65
RETREV    57
RIP        7    19    24    25    26    31    42    43    48    49    50    51
          52    53    54
RIPWID     6    31    42    53    66
SAWC       4   *37   *48   *49   *50   *51    73
SAWCUT     8   *73
SAWIND   *33   *42   *43   *44    45    47    48    49    50    51    52
SAWRIP   *31    42   *53
SAWSTO    29
SINDEX     4   *39   *52    68    70
START     57    59
STEMP    *70    71
VALSTO
```

```
  1          SUBROUTINE NOCHIP(RIPLX.RIPUX.RIPLY.RIPUY)                     NOCHIP
  2 C ***    RERIPS PIECE AS DEFINED BY ABOVE CORRDINATES; RANDOM LENGTH.   NOCHIP
  3 C ***    AND FIXED WIDTH                                                NOCHIP
  4          IMPLICIT INTEGER(A-Z)                                          NOCHIP
  5          REAL VALUE                                                     NOCHIP
  6          LOGICAL REJECT(3).FJ.FOUND                                     NOCHIP
```

```
      COMMON /MD/ NOPT,REPIP(30,2),SAL(30,4)
      COMMON /MD2/ NPEWID,PEWID(5),SALVAG(15,4,5,2)
      COMMON /MSD/ BOARD(100,24),MAXLEN,BDNUM,MINLEN
      COMMON /SD/ FJ
      COMMON /D2/ YEDGE,INDEX
      COMMON /MOD/ BDLX,BDUX,BDLY,BDUY
      DIMENSION NPIECE(2,5),OPT(30,4)
      DATA LX/1/,LY/2/,UX/3/,UY/4/,CLEAR/0/,BOTTOM/1/,TOP/2/,LEN1/1/,
     1    LEN2/2/,WID/3/
      IF((RIPUX-RIPLX).LT.MINLEN)RETURN
      IF((RIPUY-RIPLY).LE.PEWID(1))RETURN
      MINWID=REWID(1)
C *** CALCULATE WHAT WIDTHS FIT INTO THE PIECE
      NRWID=0
      DO 15 IWID=1,NPEWID
        IF(PEWID(IWID).LE.(RIPUY-RIPLY-1))NRWID=NRWID+1
   15 CONTINUE
      IF(NRWID.LE.0)RETURN
C *** TEST IF A SIDE HAS AN ALLOWABLE CLEAR LENGTH
      LOWX=RIPLX
  810 SIDE=BOTTOM
      DO 820 K=1,2
        DO815 IWID=1,5
          NPIECE(K,IWID)=0
  815   CONTINUE
  820 CONTINUE
      YEDGE=RIPLY
      INDEX=1
      FOUND=.FALSE.
  825 K2=LOWX/36
      BIT=IABS(LOWX+K2*36)
      BEG=LOWX
      END=BEG-1
      CLRKNT=0
      K2=K2+1
      K1=YEDGE-(INT(SIDE/2)-1)
      OLDTYP=FLD(BIT,1,BOARD(K1,K2))
      REJECT(LEN1)=.FALSE.
      DO 865 ILEN=LOWX,RIPUX
        IF(REJECT(LEN1))GO TO 865
        IF(FOUND)GO TO 865
        K2=ILEN/36
        BIT=IABS(ILEN-K2*36)
        K2=K2+1
        XNOW=FLD(BIT,1,BOARD(K1,K2))
        IF(XNOW.NE.OLDTYP)GO TO830
        IF(XNOW.NE.CLEAR)GO TO 855
        IF(XNOW.EQ.CLEAR)CLRKNT=CLRKNT+1
        IF(FJ.AND.CLRKNT.GE.(MAXLEN))GO TO830
        IF(ILEN.NE.RIPUX)GO TO 865
  830   IF(OLDTYP.NE.CLEAR)GO TO 860
        IF(CLRKNT.LT.MINLEN)GO TO 855
        END=ILEN
        IF(XNOW.NE.CLEAR)END=END-1
C *** LENGTH CRITERIA FULFILLED: CHECK WIDTH
        TBEG=BEG
        TEND=END
        REJECT(LEN2)=.FALSE.
        REJECT(WID)=.FALSE.
        DO 850 IWID=1,NRWID
          IF(REJECT(WID))GO TO 850
          YTEMP=YEDGE-(INT(SIDE/2)-1)
          YRANGE=YEDGE+INDEX*(PEWID(IWID)-(SIDE-1))
          REJECT(WID)=.TRUE.
          DO845 TLEN=TBEG,TEND
            IF(REJECT(LEN2))GO TO845
            K2=TLEN/36
            BIT=IABS(TLEN-K2*36)
            K2=K2+1
            WIDKNT=0
            DO835 K1=YTEMP,YRANGE,INDEX
              WIDKNT=WIDKNT+FLD(BIT,1,BOARD(K1,K2))
  835       CONTINUE
            IF((WIDKNT.EQ.0).AND.(TLEN.LT.TEND))GO TO845
            TEMLEN=TLEN-TBEG
            IF(WIDKNT.NE.CLEAR)TEMLEN=TEMLEN-1
            IF(TEMLEN.LT.MINLEN)GO TO840
C ** PREVIOUS AREA CONTAINED A CLEAR CUTTING
            NPIECE(SIDE,IWID)=NPIECE(SIDE,IWID)+1
            CALL STORE2(TBEG,TBEG+TEMLEN,IWID,NPIECE(SIDE,IWID),SIDE)
            REJECT(WID)=.FALSE.
            FOUND=.TRUE.
C *** REASSIGN AREAS
  840       IF((END-(TLEN+1)).LT.MINLEN)REJECT(LEN2)=.TRUE.
            IF(.NOT.REJECT(LEN2))TBEG=TLEN
  845     CONTINUE
          IF(REJECT(WID))GO TO 850
        N=NPIECE(SIDE,IWID)
        TBEG=SALVAG(N,LX,IWID,SIDE)
          TEND=SALVAG(N,UX,IWID,SIDE)
          REJECT(LEN2)=.FALSF.
  850   CONTINUE
  855   BEG=ILEN
  860   OLDTYP=XNOW
        CLRKNT=0
  865 CONTINUE
      INDEX=INDEX*(-1)
      IF(INDEX.GE.0)GO TO 870
      FOUND=.FALSE.
      YEDGE=RIPUY
      SIDE=TOP
      GO TO825
C *** CHOOSE MAXIMUM PIECES
  870 TIME=1
      TOPT=0
      NALL=0
      DO 920 SIDE=BOTTOM,TOP
        DO 915 K=1,NRWID
          IF(NPIECE(SIDE,K).LE.0)GO TO 915
          NALL=NALL+1
          IF(TIME.GT.1)GO TO 885
          TOPT=NPIECE(SIDE,K)
          DO 880 I=1,TOPT
            DO 875 COOR=1,4
              OPT(I,COOR)=SALVAG(I,COOR,K,SIDE)
  875       CONTINUE
  880     CONTINUE
          TIME=TIME+1
          GO TO 915
  885     NTEMP=NPIECE(SIDE,K)
          DO 910 I=1,NTEMP
            ALONE=0
            DO 900 J=1,TOPT
              DIMLEN=SALVAG(I,UX,K,SIDE)-SALVAG(I,LX,K,SIDE)
              DIMWID=SALVAG(I,UY,K,SIDE)-SALVAG(I,LY,K,SIDE)
             OPTLEN=OPT(J,UX)-OPT(J,LX)
             OPTWID=OPT(J,UY)-OPT(J,LY)
              IF(VALUE(DIMLEN,DIMWID).LE.VALUE(OPTLEN,OPTWID))GO TO 900
              DO 895 COOR=1,4
                OPT(I,COOR)=SALVAG(I,COOR,K,SIDE)
  895         CONTINUE
  900       CONTINUE
            IF(ALONE.LT.TOPT)GO TO 910
            TOPT=TOPT+1
            DO 905 COOR=1,4
              OPT(TOPT,COOR)=SALVAG(I,COOR,K,SIDE)
  905       CONTINUE
  910     CONTINUE
  915   CONTINUE
  920 CONTINUE
      IF((NALL.LE.0).AND.(ILEN.GE.RIPUX))RETURN
C *** STORE OPT PIECE IN RERIP AND SALVAGE COORDINATES IN
```

```
149 C ***  SALVAG(N,M,1,1)                                         NOCHIP
150        DO 935 I=1,TOPT                                         NOCHIP
151          REPIP(NOPT+I,1)=OPT(I,UX)-OPT(I,LX)                   NOCHIP
152          REPIP(NOPT+I,2)=OPT(I,UY)-OPT(I,LY)                   NOCHIP
153          DO 925 COOP=1,4                                       NOCHIP
154            SAL(NOPT+I,COOR)=OPT(I,COOR)                        NOCHIP
155    925   CONTINUE                                              NOCHIP
156          TY=SAL(NOPT+I,UY)                                     NOCHIP
157          BY=SAL(NOPT+I,LY)                                     NOCHIP
158          BX=SAL(NOPT+I,LX)                                     NOCHIP
159          TX=SAL(NOPT+I,UX)                                     NOCHIP
160          IF(BX.GT.BDLX)CALL STORE(BX-1,BX,BY-1,TY)             NOCHIP
161          IF(TX.LT.BDUX)CALL STORE(TX,TX+1,BY-1,TY)             NOCHIP
162 C *** REPIP SAWCUTS                                             NOCHIP
163          IF(BX.LE.BDLX)BX=BDLX+1                               NOCHIP
164          IF(TY.NE.RIPUY)GO TO 930                              NOCHIP
165          IF(BY.GT.BDLY)CALL STORE(BX-1,TX+1,BY-1,BY)           NOCHIP
166          GO TO 935                                             NOCHIP
167    930   IF(TY.LT.BDUY)CALL STORE(BX-1,TX+1,TY,TY+1)           NOCHIP
168    935 CONTINUE                                                NOCHIP
169        NOPT=NOPT+TOPT                                          NOCHIP
170        LOWX=SAL(NOPT,UX)+1                                     NOCHIP
171        BEG=LOWX                                                NOCHIP
172        IF((RIPUX-BEG).LT.MINLEN)RETURN                         NOCHIP
173        GO TO810                                                NOCHIP
174        END                                                     NOCHIP

                      ***  STATEMENT NUMBERS  ***

   15     21   *23
  810    *27   173
  815     29   *31
  820     28   *32
  825    *36   108
  830     52    55   *57
  835     77   *79
  840     83   *90
  845     71    72    80   *92
  850     66    67    93   *98
  855     53    58   *99
  860     57  *100
  865     45    46    47    56  *102
  870    104  *110
  875    120  *122
  880    119  *123
  885    117  *126
  895    135  *137
  900    129   134  *138
  905    141  *143
  910    127   139  *144
  915    114   115   125  *145
  920    113  *146
  925    153  *155
  930    164  *167
  935    150   166  *168

                      ***  VARIABLES  ***

ALONE   *128   139
BDLX      12   160   163
BDLY      12   165
BDNUM      9
BDUX      12   161
BDUY      12   167
BEG      *38    39    62   *99  *171   172
BIT      *37    43   *49    51   *74    78
BOAPD      9    43    51    78
BOTTOM   *14    27   113
BX      *158   160  *163   165   167
BY      *157   160   161   165
CLEAP    *14    53    54    57    60    82
CLPENT   *40   *54    55    58  *101
COOP    *120   121  *135   136  *141   142  *153   154
DIMLEN  *130   134
DIMWID  *131   134
END      *39   *59   *60    63    90
FJ         6    10    55
FLD       43    51    78
FOUND      6   *35    47   *88  *105
I       *119   121  *127   130   131   136   142  *150   151   152   154   156
         157   158   159
IABS      37    49    74
ILEN     *45    48    49    56    59    99   147
INDEX     11   *34    69    77  *103   104
INT       42    68
IWID     *21    22   *29    30   *66    69    85    86    94    95    96
J       *129   132   133
K        *28    30  *114   115   118   121   126   130   131   136   142
K1       *42    43    51   *77    78
K2       *36    37   *41    43   *48    49   *50    51   *73    74   *75    78
LEN1     *14    44    46
LEN2     *14    64    72    90    91    97
LOWX     *26    36    37    38    45  *170   171
LX       *14    95   130   132   151   158
LY       *14   131   133   152   157
MAXLEN     9    55
MINLEN     9    16    58    83    90   172
MINWID   *18
N        *94    95    96
NALL    *112  *116   147
NOCHIP     1
NOPT       7   151   152   154   156   157   158   159  *169   170
NPIECE    13   *30   *85    86    94   115   118   126
NREWID     8    21
NRWID    *20   *22    24    66   114
NTEMP   *126   127
OLDTYP   *43    52    57  *100
OPT       13  *121   132   133  *136  *142   151   152   154
OPTLEN  *132   134
OPTWID  *133   134
REJECT     6   *44    46   *64   *65    67   *70    72   *87   *90    91    93
         *97
RERIP      7  *151  *152
REWID      8    17    18    22    69
RIPLX      1    16    26
RIPLY      1    17    22    33
RIPUX      1    16    45    56   147   172
RIPUY      1    17    22   106   164
SAL        7  *154   156   157   158   159   170
SALVAG     8    95    96   121   130   131   136   142
SIDE     *27    42    68    69    85    86    94    95    96  *107  *113   115
         118   121   126   130   131   136   142
STORE    160   161   165   167
STORE2    86
TBEG     *62    71    81    86   *91   *95
TEMLEN   *81   *82    83    86
TEND     *63    71    80   *96
TIME    *110   117  *124
TLEN     *71    73    74    80    81    90    91
TOP      *14   107   113
TOPT    *111  *118   119   129   139  *140   142   150   169
TX      *159   161   165   167
TY      *156   160   161   164   167
UX       *14    96   130   132   151   159   170
UY       *14   131   133   152   156
VALUE      5   134
WID      *14    65    67    70    87    93
WIDKNT   *76   *78    80    82
XNOW     *51    52    53    54    60   100
YEDGE     11   *33    42    68    69  *106
YRANGE   *69    77
YTEMP    *68    77
```

```
 1          SUBROUTINE STORE(LX,UX,LY,UY)                              STORE
 2          IMPLICIT INTEGER(A-Z)                                      STORE
 3          LOGICAL FOUND                                              STORE
 4          COMMON /MHOV/SAWCUT(100,4),AVAIL                           STORE
 5          COMMON /OSCHP/FOUND                                        STORE
 6          IF(AVAIL.LE.100) GO TO 1050                                STORE
 7          WRITE(6,1010)                                              STORE
 8   1010   FORMAT(' MORE THAN 100 SAWCUTS')                           STORE
 9          RETURN                                                     STORE
10   1050   SAWCUT(AVAIL,1)=LX                                         STORE
11          SAWCUT(AVAIL,2)=UX                                         STORE
12          SAWCUT(AVAIL,3)=LY                                         STORE
13          SAWCUT(AVAIL,4)=UY                                         STORE
14          AVAIL=AVAIL+1                                              STORE
15          IF(.NOT.FOUND)CALL SAWSTO(LX,UX,LY,UY)                     STORE
16          RETURN                                                     STORE
17          END                                                        STORE
```

*** STATEMENT NUMBERS ***

```
1010       7    *8
1050       6   *10
```

*** VARIABLES ***

```
AVAIL      4     6    10    11    12    13   *14
FOUND      3     5    15
LX         1    10    15
LY         1    12    15
SAWCUT     4   *10   *11   *12   *13
SAWSTO    15
STORE      1
UX         1    11    15
UY         1    13    15
```

```
 1          SUBROUTINE STORE2(BEG,END,WIDTH,NPIECE,SIDE)               STORE2
 2          IMPLICIT INTEGER(A-Z)                                      STORE2
 3 C ***    STORES SALVAGE PIECES                                      STORE2
 4          COMMON /MD2/NREWID,REWID(5),SALVAG(15,4,5,2)               STORE2
 5          COMMON /D2/YEDGE,INDEX                                     STORE2
 6          DATA BOTTOM/1/,TOP/2/,LX/1/,LY/2/,UX/3/,UY/4/              STORE2
 7          IF(NPIECE.LE.15)GO TO1110                                  STORE2
 8          WRITE(6,1105)                                              STORE2
 9   1105   FORMAT(10X,'MORE THAN 15 SALVAGE PIECES PER RIP')          STORE2
10          RETURN                                                     STORE2
11   1110   SALVAG(NPIECE,LX,WIDTH,SIDE)=BEG                           STORE2
12          SALVAG(NPIECE,UX,WIDTH,SIDE)=END                           STORE2
13          IF(INDEX)1120,1140,1130                                    STORE2
14   1120   SALVAG(NPIECE,LY,WIDTH,SIDE)=YEDGE-REWID(WIDTH)            STORE2
15          SALVAG(NPIECE,UY,WIDTH,SIDE)=YEDGE                         STORE2
16          RETURN                                                     STORE2
17   1130   SALVAG(NPIECE,LY,WIDTH,SIDE)=YEDGE                         STORE2
18          SALVAG(NPIECE,UY,WIDTH,SIDE)=YEDGE+REWID(WIDTH)            STORE2
19   1140   RETURN                                                     STORE2
20          END                                                        STORE2
```

*** STATEMENT NUMBERS ***

```
1105       8    *9
1110       7   *11
1120      13   *14
1130      13   *17
1140      13   *19
```

*** VARIABLES ***

```
BEG        1    11
BOTTOM    *6
END        1    12
INDEX      5    13
LX        *6    11
LY        *6    14    17
NPIECE     1     7    11    12    14    15    17    18
NREWID     4
REWID      4    14    18
SALVAG     4   *11   *12   *14   *15   *17   *18
SIDE       1    11    12    14    15    17    18
STORE2     1
TOP       *6
UX        *6    12
UY        *6    15    18
WIDTH      1    11    12    14    15    17    18
YEDGE      5    14    15    17    18
```

```
 1          FUNCTION VALUE(LENGTH,WIDTH)                               VALUE
 2 C ***    FUNCTION TO FIND THE VALUE OF GIVEN PIECE                  VALUE
 3          IMPLICIT INTEGER(A-Z)                                      VALUE
 4          REAL VALUE                                                 VALUE
 5          COMMON /MW/INDEX(10,5),LVALUE(10),UVALUE(5)                VALUE
 6          VALUE=0                                                    VALUE
 7 C ***    MATCH WIDTH                                                VALUE
 8          WINDEX=0                                                   VALUE
 9          DO 970 IWID=1,5                                            VALUE
10            IF(WINDEX.GT.0)GO TO 970                                 VALUE
11            IF(WIDTH.LE.UVALUE(IWID))WINDEX=IWID                     VALUE
12    970   CONTINUE                                                   VALUE
13 C ***    MATCH LENGTH                                               VALUE
14          LINDEX=0                                                   VALUE
15          DO 980 LWID=1,9                                            VALUE
16            IF(LINDEX.GT.0)GO TO 980                                 VALUE
17            IF(LWID.GE.9)LINDEX=9                                    VALUE
18            IF((LENGTH-3).LE.(4*LVALUE(LWID)))LINDEX=LWID            VALUE
19    980   CONTINUE                                                   VALUE
20          VALUE=FLOAT(INDEX(LINDEX,WINDEX)*.001)*                    VALUE
21        1      FLOAT(LENGTH*WIDTH)/(16.*144.)                        VALUE
22          IF(INDEX(LINDEX,WINDEX).EQ.1)VALUE=VALUE*1000.             VALUE
23          RETURN                                                     VALUE
24          END                                                        VALUE
```

*** STATEMENT NUMBERS ***

```
 970       9    10   *12
 980      15    16   *19
```

*** VARIABLES ***

```
FLOAT     20
INDEX      5    20    22
IWID      *9    11
LENGTH     1    18    20
LINDEX   *14    16   *17   *18    20    22
LVALUE     5    18
LWID     *15    17    18
VALUE      1     4    *6   *20   *22
WIDTH      1    11    20
WINDEX    *8    10   *11    20    22
UVALUE     5    11
```

Appendix E: Listing of Computer Input and Output Used in Processing the Sample Board (In Fig. 1)

```
RANDOM   =YES
NOPRINT  =NO
DOLMAX   =YES
EDGING   =1
NSAWS    =8
NWIDTH   =5
NREWIDTH =5
NLEN     = 1
SAWMIN   = 9.0
SAWMAX   = 84.0
      2.50      3.00      3.50      4.25      4.75
      1.75      2.50      3.00      3.50      4.25
   9.0
      12       19       26       35       47       59       71       83       84
       7        9       13       17       19
     780      790      800      820      840      860      880     1000     1050
     800      810      820      840      860      880     1000     1030     1150
     810      830      840      860      880      910     1040     1070     1200
     820      840      850      870      890      930     1080     1120     1250
     825      845      855      875      895      950     1100     1145     1300
01306 XX SHOP  6/4 PONDEROSA PINE       DEFECT TOTAL  19   00 000 47 677      01
01306 01     + + 14 14 6/4 POND PINE 11 3 14 SAMPLE BOARD       UNIT 013225   02
01306     WN00 -  000 30 -  008                                             1 03
01306     WN00 04 238 00 05 290 NK00 -  290 16 -  334 WN43 44 089 44 47 113  1 04
01306     WN00 13 000 00 02 118 SP40 47 113 37 47 261 SP37 47 261 40 47 323  1 05
01306     SP30 -  323 47 -  362 SP31 47 362 36 47 422 SP36 -  422 37 -  497  1 06
01306     SP32 47 497 30 47 513 SP30 47 513 33 47 560 SP33 47 560 36 47 583  1 07
01306     NK07 -  406 16 -  425 NK29 -  609 39 -  624 SP28 47 624 18 47 643  1 08
01306     SP31 47 643 36 47 658 NK18 -  658 47 -  677 SC12 -  673 18 -  677  1 09
END
          STOP
```

Figure 13.—Listing of input to process sample board shown in figure 1.

OPTYLD

EDGING ALLOWANCE (IN.)	.25
CUTTING LENGTH OPTION	RANDOM
RIP WIDTHS (IN.)	2.50 3.00 3.50 4.25 4.75
SALVAGE WIDTHS (IN.)	1.75 2.50 3.00 3.50 4.25
MINIMUM CUTTING LENGTH (IN.)	9.00
MAXIMUM CUTTING LENGTH (IN.)	84.00
SALVAGE LENGTHS	RANDOM
VALUE MAXIMIZATION	

VALUE TABLE

	12	19	26	35	47	59	71	83	84	
1.75"		780	790	800	820	840	860	880	1000	1050
2.25"		800	810	820	840	860	880	1000	1030	1150
3.25"		810	830	840	860	880	910	1040	1070	1200
4.25"		820	840	850	870	890	930	1080	1120	1250
4.75"		825	845	855	875	895	950	1100	1145	1300

Figure 14.—Listing of computer output when sample board (fig. 1) was processed.

```
 XX SHOP   BOARD NUMBER   1306

  LENGTH =  169.25  WIDTH =  11.75

PRIMARY CUTTINGS                       VALUE
             30.00   X  3.50           .63
             18.00   X  3.50           .37
             62.00   X  3.50          1.63
             84.00   X  3.50          2.55
             66.00   X  3.50          1.73
             28.25   X  3.00           .51
    SURFACE AREA = 994.75   TOTAL VALUE=  7.42
         % YIELD =  50.02

RERIP CUTTINGS                         VALUE
             17.00   X  1.75           .16
             12.75   X  2.50           .18
             30.75   X  1.75           .31
             12.75   X  1.75           .12
    SURFACE AREA = 137.75   TOTAL VALUE=   .77
         % YIELD =   6.93

               BOARD AREA   =   1988.69  SQ.IN.
               TOTAL YIELD =      56.95  PERCENT
               VALUE                8.19


    SAWCUT COORDINATES LX UX LY UY
       0       677       0        1
       0       677      15       16
      48        49       7       15
     117       118       7       15
```

Figure 14.—Listing of computer output when sample board (fig. 1) was processed—cont.

```
     48        118          7         8
    117        118          1        15
    238        239          4        15
    290        291          4        15
    238        291          4         5
    238        239          1        15
    333        334          1        15
    406        407          1        15
    424        425          1        15
    673        674          1        15
      0        677         30        31
      7          8         16        30
    344        345         16        30
    609        610         16        30
      0        677         43        44
    113        114         30        38
    237        238         30        38
    113        238         38        39
    271        272         30        38
    323        324         30        38
    271        324         38        39
    113        114         31        43


  SALVAGE COORDINATES LX UX LY UY
     49          8        117        15
    239          5        290        15
    114         31        237        38
    272         31        323        38
**** TIME AT END    IS        4.187 SECONDS *****

    STOP
```

U.S. Forest Products Laboratory SA 6-1

OPTYLD--A multiple rip-first computer program to maximize cutting yields, by Pamela J. Giese and Kent A. McDonald. Madison, Wis., FPL, 1982

33 p. (USDA For. Sérv. Res. Pap. FPL 412)

A computer program designed to simulate multiple rip-first, then crosscut, then rerip operations. These operations are practiced by the softwood moulding and millwork industries.

The program is written FORTRAN V.

KEYWORDS: lumber utilization, cutting yields, softwood millwork, optimum yields, crosscut, rerip.

1-8

DTI